# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*

**1. REPORT DATE (DD-MM-YYY)**
21-February-2009

**2. REPORT TYPE**
Final Report

**3. DATES COVERED** *(From – To)*
01-Jul-2009 to 21-Feb-2010

**4. TITLE AND SUBTITLE**
INTEGRATED RECONFIGURABLE INTELLIGENT SYSTEMS (IRIS) FOR COMPLEX NAVAL SYSTEMS

**5a. CONTRACT NUMBER**
N00014-09-C-0394

**5b. GRANT NUMBER**
N/A

**5c. PROGRAM ELEMENT NUMBER**
N/A

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**

**6. Author(s)**
Dr. Dimitri N. Mavris

Dr. Yongchang Li

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Georgia Institute of Technology
School of Aerospace Engineer
Atlanta, GA 30332-0150

**8. PERFORMING ORGANIZATION REPORT NUMBER**
N/A

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
Office of Naval Research
875 North Randolph Street
Arlington, VA 22203-1995

**10. SPONSOR/MONITOR'S ACRONYM(S)**
ONR

**11. SPONSORING/MONITORING AGENCY REPORT NUMBER**
N/A

**12. DISTRIBUTION AVAILABILITY STATEMENT**
Unlimited Distribution

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**

To meet the requirements for next generation naval ship, more emphasis has been given to increasing survivability and mission effectiveness and reducing operating cost through increased automation and intelligence. The Aerospace Systems Design Laboratory (ASDL) at the Georgia Institute of Technology proposed a framework referred to as Integrated Reconfigurable Intelligent System (IRIS) for facilitating the design and operation of naval complex systems. The following report details the progress that has been made by ASDL in developing and applying the IRIS concept for the period of July 1 2009 to February 21, 2010. The team attempted to develop a design process for intelligent complex system utilizing Unified Modeling Language and addressed how the developed modeling and simulation environment can support this design process. In addition, the integrated modeling and simulation environment was refined and further developed. Progress has been made on individual tasks associated with model development and integration improvement.

**15. SUBJECT TERMS**
Modeling & Simulation, Reconfigurability, Integrated & Intelligent, Naval Systems

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | Mr. ANTHONY J. SEMAN |
| | | | | | **19b. TELEPHONE NUMBER** *(include area code)* |
| U | U | U | SAR | 80 | 703-696-5992 |

# FINAL REPORT

**(July 1, 2009 – February 21, 2009)**

# "Integrated Reconfigurable Intelligent Systems (IRIS) for Complex Naval Systems"

**Contract #: N00014-09-C-0394**

**SUBMITTED TO:**
**Mr. Anthony J. Seman, Office of Naval Research**
**(email: Anthony_Seman@onr.navy.mil)**

**Office of Naval Research**
**875 North Randolph Street**
**Arlington, VA 22203-1995**

**SUBMITTED BY:**
Georgia Institute of Technology
School of Aerospace Engineering
Aerospace Systems Design Laboratory (ASDL)
Atlanta, Georgia 30332-0150

**AWARD PERIOD:**
**July 1, 2009 to February 21, 2009**

**February 21, 2010**

**Principal Investigator:**
Professor Dimitri N. Mavris
Director
Aerospace Systems Design Laboratory
School of Aerospace Engineering
Georgia Institute of Technology
Phone: (404) 894-1557
dimitri.mavris@ae.gatech.edu

# TABLE OF CONTENT

# Summary

To meet the requirements for next generation naval ship, more emphasis has been given to increasing survivability and mission effectiveness and reducing operating cost through increased automation and intelligence. The Aerospace Systems Design Laboratory (ASDL) at the Georgia Institute of Technology proposed a framework referred to as Integrated Reconfigurable Intelligent System (IRIS) for facilitating the design and operation of naval complex systems. The following report details the progress that has been made by ASDL in developing and applying the IRIS concept for the period of July 1 2009 to February 21, 2010. The team attempted to develop a design process for intelligent complex system utilizing Unified Modeling Language and addressed how the developed modeling and simulation environment can support this design process. In addition, the integrated modeling and simulation environment was refined and further developed. Progress has been made on individual tasks associated with model development and integration improvement. New achievements for this period include:

- Created UML diagrams (use case, activity, communication, class diagrams) for developing a design process for intelligent complex system
- Formulated an integrated design process in which  stakeholders and required methods/tools at each design step are identified
- Developed the notional-YP geometry model using Paramarine and exported to CAD files that can be imported to Rhino
- Improved the integration scheme and conducted error evaluation for the simulation environment
- Implemented and integrated the high level controller for solving the resource allocation problem
- Implemented the distributed dynamic probabilistic inference engine to the YP model and integrated to the simulation environment
- Developed the graph-based model of the notional-YP cooling system and developed and tested the reference damage controller
- Developed a new version of the HMI, documented the interface, and created script for integrating with other models
- Developed a baseline naval architecture to demonstrate the theoretical framework for survivability design
- Finished 3 journal papers which are under review or preparation

# Task 1: Design of Integrated Heterogeneous Systems

## *Subtask 1.1: Design Process Development Using System Engineering Approaches*

### Subtask 1.1.1: Method Development for Complex System Design

### Introduction

The success of a product design often depends on the process used to design the product, consequently, great efforts have been made to develop advanced design processes which will ultimately lead to a product with increased performance and reduced cost. It is observed that if the process is fully understood and improved before a project is started, the outcome will be achieved in a more efficient and effective manner. However, many design processes were developed in an empirical way (often through a brain storming) and represented by a static flow chart. It is usually difficult for a designer to implement such developed design process since it often requires the designer to figure out the essentials of each design step of the process, such as where to start, what should be completed, who needs to be involved at each design process. Therefore, a systematic approach needs to be utilized to design a process that can bring all this information together and lead to a final product with high benefit to cost ratio. An approach is proposed in this research for developing an advanced design process which can be used to design an intelligent and complex system.

The objective of this work is to represent the design process of a class of products by employing a systematic approach and support process development within the general discipline of design using UML (Unified Modeling Language). However, it is observed that in general a step-by-step process for utilizing UML for process modeling does not exist, since the modeling process is dependent on the problem and the starting point. For example, while developing software solution to a problem it is often the case that some core libraries or class will be the foundation for a starting point of the project. This is consistent with the concepts of reusable code. Given this scenario the designers might proceed by first representing the initial foundation of the system in order to build up to final product. However, in this example the starting point is less obvious and thus would lead to a different approach. In addition, in the design of large scale systems such as naval ship, heterogeneous systems interact with each other to exhibit a complex behavior, which is difficult to be addressed in the traditional design process. Thus, a systematic approach is needed to model and develop advanced process for complex systems design so the process can be employed to produce design solution that is ultimately successful.

## Process Modeling Using UML

In this study, a modeling process using UML will be presented for designing intelligent complex systems, more specifically, those that fall under the category of IRIS (integrated reconfigurable intelligent systems) systems. As discussed before, an IRIS designed system is envisioned to possess the functionalities of assessing, predicting, planning and executing. The implementation of these functions involves the extensive use of autonomous decision making to deal with various scenarios. Traditional methods fall short of adequately addressing all the capability requirements within the stipulations of an acceptable cost. Consequently, new design processes must be developed, implementing the methods and tools necessary to design systems with respect to vital global behaviors.

A generic modeling process is proposed to formulate the design process for intelligent complex system. The modeling process will capture all information necessary for a design process, such as what expertise is needed, what activities need to be completed, what method/tools need to be used, what models need to be developed and how the models interact with each other. In addition, the modeling process will be able to facilitate the tracking of changes in the design process, for example, when the requirements change, the process will be capable of effectively tracking and accommodating the changes.

The final product of this modeling process is a design process. UML is ideal for capturing process and illustrating them for a general engineering audience. It is considered as an appropriate approach to describe processes and has been widely used in business process modeling. As an extension to the business process modeling concept, UML is being employed to formulate the IRIS design process. Unlike other traditional design processes, the IRIS design process will address multiple necessary interface actors such as the stakeholders at each design step, the utilization of resources, interdependency between different steps, interrelations between models, methods and tools required at each design step and so on. That is, the IRIS design process put more emphasis on how the activities are accomplished rather than what needs to be accomplished at each design step. This offers more useful information for designer and will highly increase the design efficiency and the probability of success of the final design.

### ***Use Case Diagram***

The process starts by creating a use case diagram to capture the functionality and requirements of the IRIS design process. The use case diagram in most cases is the first diagram considered during a modeling exercise with UML. Simply it is because the first step to any design exercise is to gain an understanding of the problem definition. The use case diagram is a big picture showing the behavior of the system and describes how 'actors' expect to use the system. For example, considering a word processor, the drivers of the design will depend on the intended use of the system. If the users or 'actors' intend to use the system for drafting and publishing form letters, then the design of the system will require the necessary components for this use case. It is also possible depending on the frequency of each use case that the overall system might be more tailored towards some use cases more than others when confronted with design tradeoffs. For modeling the IRIS design process, a use case diagram is developed, where four high level use cases and their dependencies are defined (build system ability set, design architecture concept, establish baseline and build system model), and the actors/stakeholders (such as program manager, system architect, sub-system modeler, sub-system analysis) interacting with these uses cases are identified as well, as shown in Figure 1**Error! Reference source not found.**.

In this example, each use case represents a functionality that the design process provides to produce physical system designs. In addition, the use case diagram describes how the stakeholders expect to use the design process. It is vital that this step captures the organizational characteristics and responsibilities of each stakeholder, thus it can be known that what expertise is needed at each design process and how they collaborate with each other.

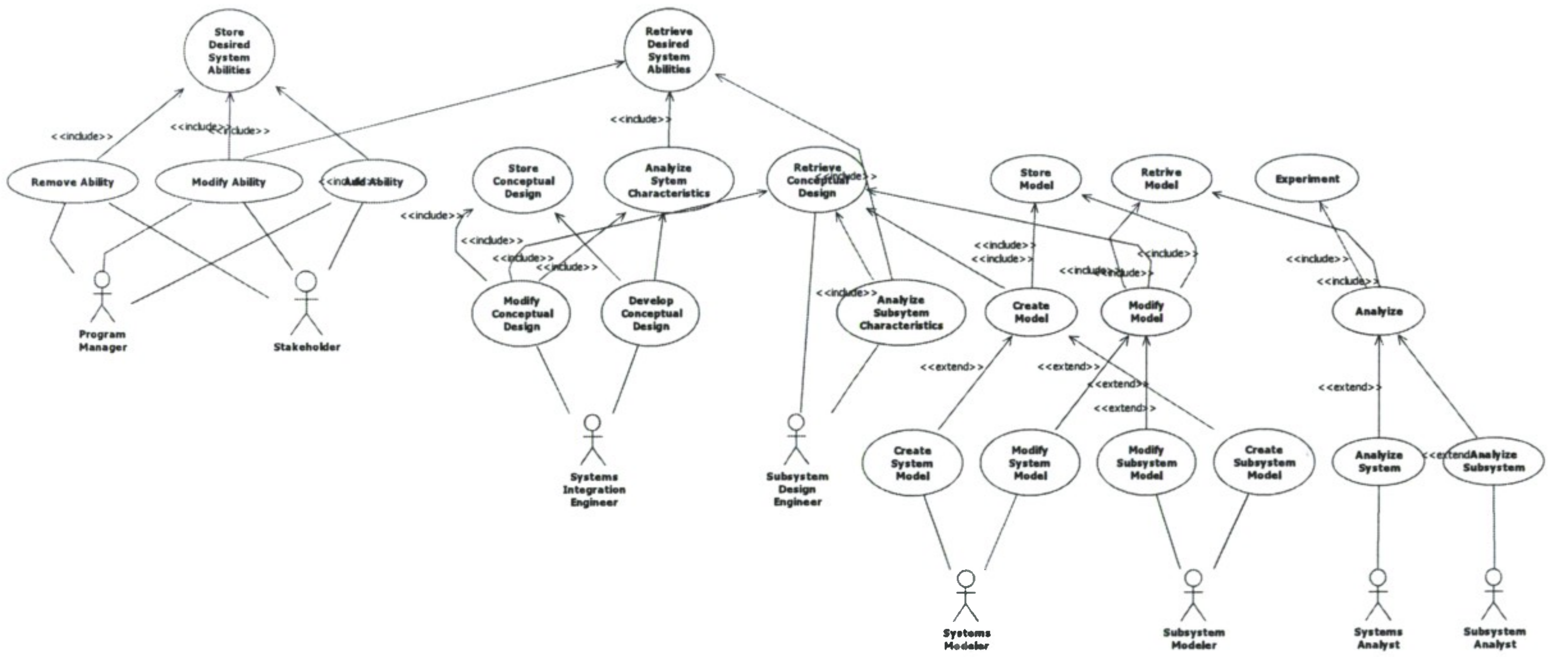


Figure 1: Use Case Diagram for Design Process

### Activity Diagram

Use case diagram provides a graphical overview of the functionalities of the design process and the interactions between the process and the stakeholders. To further understand the design process, the use cases need to be studied and analyzed. This can be done through creating the activity diagrams which describe the workflow behavior of a system and model the logic captured by a use case. Activity diagram is like a conventional flowchart and allows parallel activities to be modeled. It is presented in a relatively more detailed manner, one can observe all the dependencies between activities in a graphic form, and enhance the ability to spot where changes and improvements must be done. One or more activity diagrams are created for each high level use case to model the logic supporting the use case or usage scenario and show the overall work flow. Since the nature of this process, originally developed activity diagrams are refined several times in order to better reflect the workflow of the use case.

Figure 2 presents the refined activity diagram for "build system ability set" use case. As it can be seen, in the refined version of activity diagram, swimlanes (horizontal or vertical lane grouping a subset of activities) are applied to show more information about the workflow. Originally, swimlane is used to depict what or who is working on a particular subset of activities. However, in this implementation, swimlane is used to indicate how a particular subset of process is done. That is, what kinds of tools or methods are required to conduct those activities. This will lead us to identify the characteristics of the tool and methods required at each design step. Another six activity diagrams are developed and presented in the Appendix.

### Communication Diagram

Activity diagrams deliver rich information about the workflow of a use case or usage scenario of the design process. However, it is not capable of showing the detailed interactions and message flows between objects. This information can be represented in communication diagram in which it combine the information from class, sequence and use case diagrams to describe both the static structure and dynamic behavior of the process. We also created communication diagrams for each use cases. Figure 3 shows the communication diagram for "build system ability set" use case. Other developed communication diagrams are presented in Appendix.

### Class Diagram

In the communication diagrams, the objects of the design process are defined first and then the interactions between the objects are illustrated. An object is an instance of a class, and can be created from that class. To better describe the relations between objects, a class diagram is needed since a class diagram can describe the types of objects the design process has and show its static structure.

In this implementation, a high level class diagram is created to identify the methods and tools that are required at each design step based on the developed activity and communication diagrams. From the activity diagram, *how* to conduct a set of process is identified through the defined swimlane. In addition, from the communication diagram,

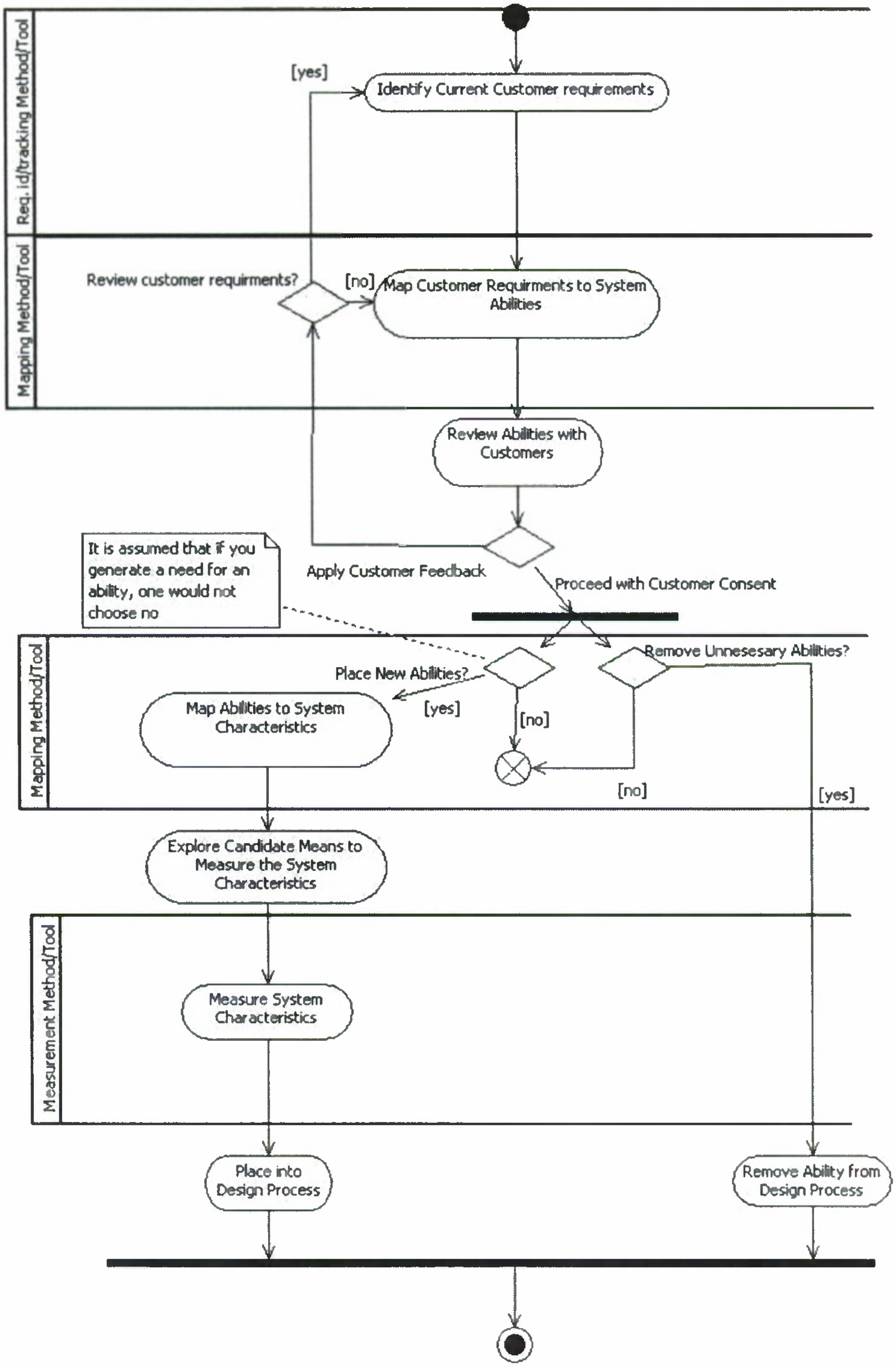

Req. id/tracking Method/Tool
[yes]
Identify Current Customer requirements
Mapping Method/Tool
Review customer requirments?
[no]
Map Customer Requirments to System Abilities
Review Abilities with Customers
Apply Customer Feedback
Proceed with Customer Consent
It is assumed that if you generate a need for an ability, one would not choose no
Mapping Method/Tool
Place New Abilities?
Remove Unnesesary Abilities?
Map Abilities to System Characteristics
[yes]
[no]
[no]
[yes]
Explore Candidate Means to Measure the System Characteristics
Measurement Method/Tool
Measure System Characteristics
Place into Design Process
Remove Ability from Design Process

**Figure 2: Activity Diagram for "Build System Ability Set"**

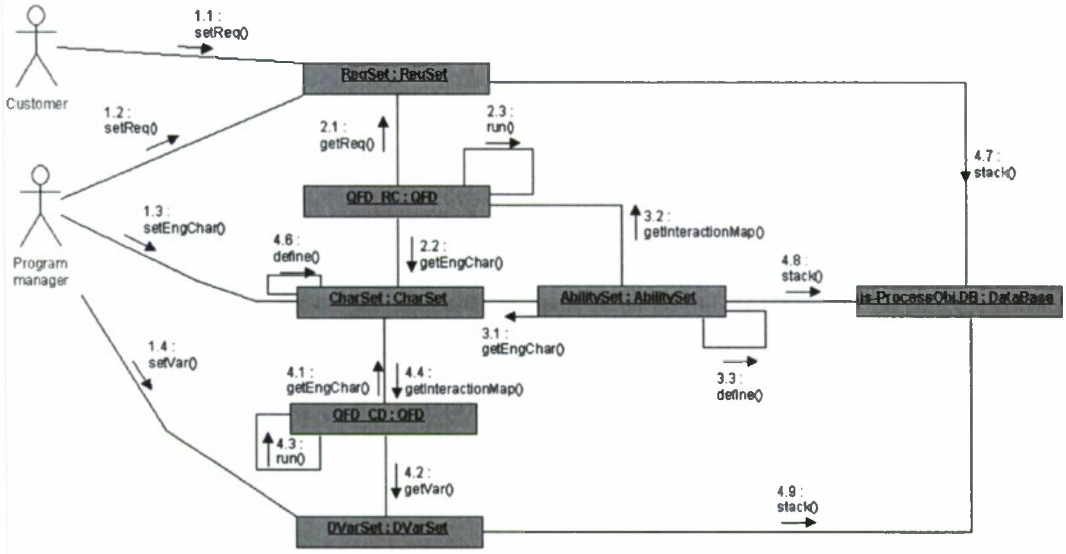


**Figure 3: Communication Diagram for "Build System Ability Set"**

the objects composed in the design process are identified as well. The class diagram is developed to further address this information, as shown in Figure 4. As it can be seen, seven major classes and their dependencies are defined to represent the necessary methods and tools needed in the design process. These class diagrams can be utilized to identify the appropriate method/tool with the necessary characteristics defined as attributes and operations in each of class. For example, for the class of "strategic decision making method/tool", a method is needed to help high level project management with making good decision. The method/tool is required to identify the requirements, allocate the limited budget to multiple projects/tasks based on the requirement prioritization and risk analysis. If multiple such methods/tools are existing, the most appropriate one will be chosen to conduct the design activity based on the attributes of the class, such as its availability, cost and ease of use.

## *Integrated Design Process*

After the activity, communication and class diagrams are created, the information presented in these diagrams is integrated together based on the use case diagram (Figure 1) and reorganized as a design process, as shown in Figure 5. As can be seen from Figure 5, the integrated design process consists of eight steps. The stakeholders involve in each design step are depicted and the candidate methods/tools required at each step are listed in the figure as well. The process starts through defining CONOPS (Concept of Operations), allocating budget based on the defined CONOPS and assessing the possible risk by high level management. Then, the program manager and the customer work together to identify the customer requirements and decompose the functionalities of the system (IRIS designed system is envisioned to posses the four main functions: assess, predict, plan and execute). Sequentially, system architect will define the system architecture based on the system and sub-system functional decomposition. Then system engineer will utilize morphological matrix kind of method to generate concept space then select one concept based on which baseline can be created. With the defined baseline, a modeling and simulation environment will be developed for design space analysis and exploration. In this step, the feasibility and viability of the system will be evaluated based on the defined evaluation matrix (in this implementation, four high level evaluation matrices are defined: robustness, resilience, mission effectiveness and operating cost). Then optimization will be conducted based on the feasible design in order to select the design which can best meet the customer requirements, which can be done by using the MADM (multi-attribute decision making) method. Finally, the design will be tested, if successful, and will be implemented.

## Future Work

Future work regarding this task will be to further improve and integrate the UML diagrams. In addition, the integrated design process will be refined as well so that it can present as much information as possible to help designer to design an intelligent complex system. Sequentially, the integrated diagrams will be used to describe and represent the new advanced design method for intelligent systems. With the employment of UML diagram, the new design process will help designer with identifying the key design requirements and activities, developing the necessary tools and methods that are required to complete each activity, and understanding the interactions among the design activities.

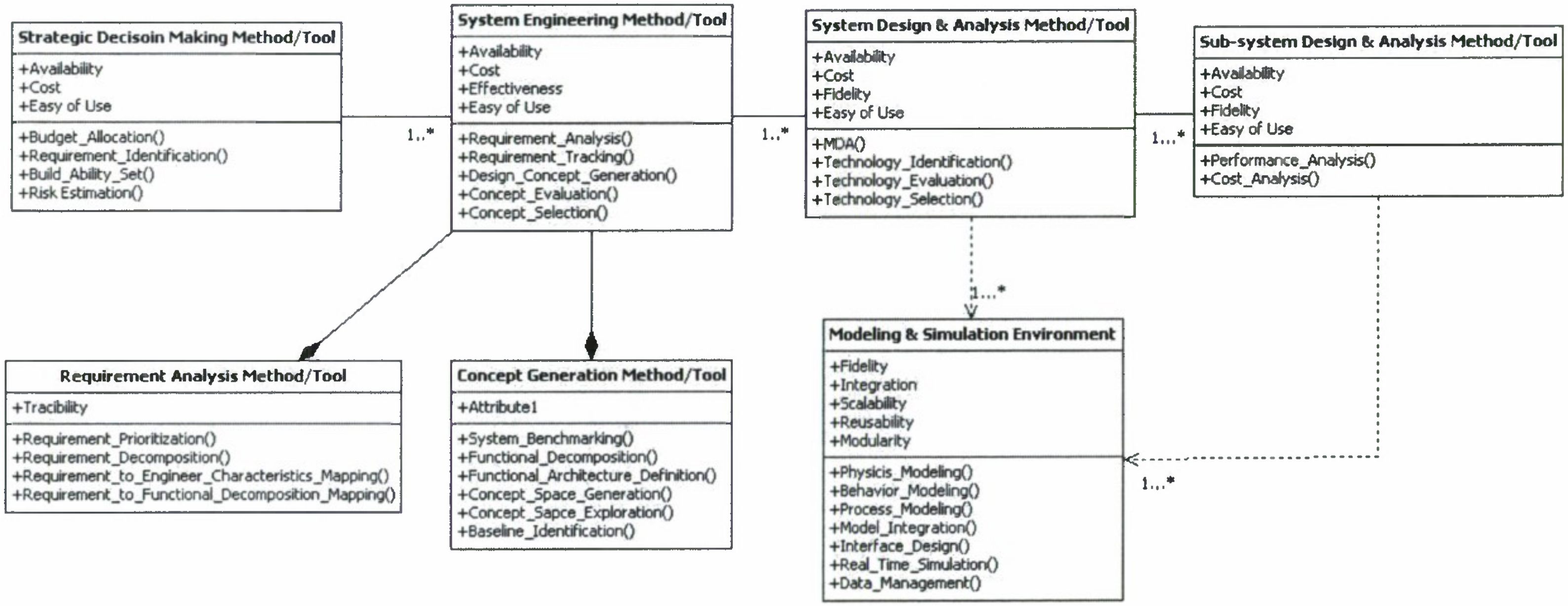


**Figure 4: Class Diagram for Design Process**

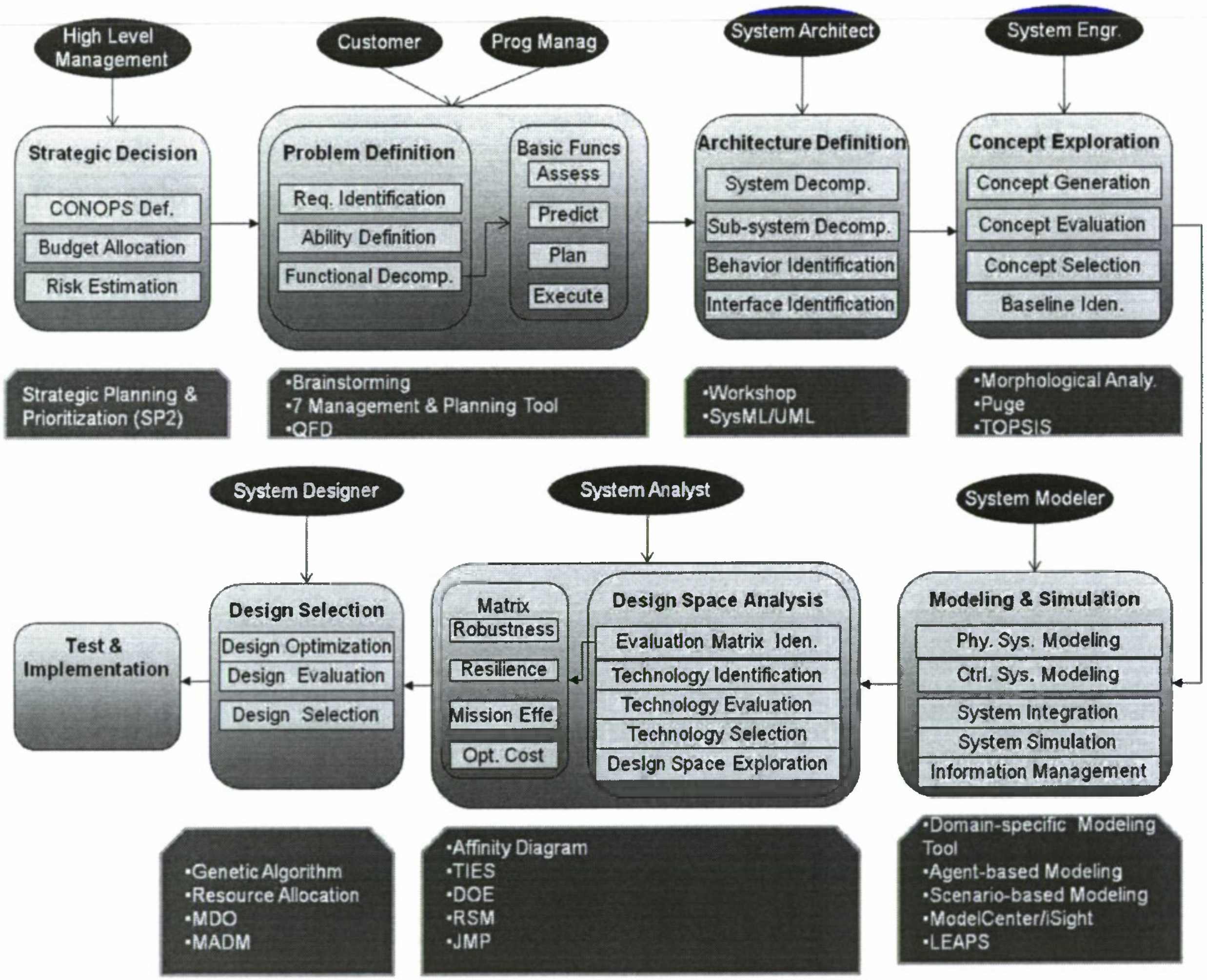


**Figure 5: Integrated Design Process**

## Subtask 1.1.2: Notional Ship Development

### Introduction

Ship system performance analysis is a critical enabler for conducting survivability studies, as well as a process step for developing and testing ship designs. Most software tools that are available for this purpose and are used either by government naval agencies or private shipbuilding companies are primarily offering functionalities, such as ship sizing, performance and cost analysis, requirements auditing, all under a CAD-based visualization environment. One of them is Paramarine®, by Graphics Research Corporation (GRC), that is a ship sizing and synthesis tool it has been recently acquired by ASDL. Besides its strong capability in CAD visualization, Paramarine carries many possibilities for conducting various types of analysis related to naval architecting. Some of the most common are stability analysis, ship weight estimation and sizing, system health monitoring, finite element analysis, etc.

In order to demonstrate the capabilities of an IRIS ship, a ship baseline model is necessary and with all the analysis possibilities of Paramarine, a decision has been made to implement a ship baseline in this new environment. A Yard Patrol (YP) craft had been proposed in the past as a suitable baseline for demonstrating and testing the advanced features of an IRIS ship. This baseline environment that is based on a notional YP-679, includes the ship geometry, including the architecture along with the internal subsystem distribution. Regarding the latter, it has been instructed that a systems architecture that resembles ONR's Tabletop Simulator (v2.0) would be an appropriate layout that can also be compatible to the YP configuration. Therefore, a notional systems architecture that mimics the simulators network has been proposed and implemented in the virtual ship model, including both the power generation, distribution and cooling systems.

### Geometry Model Development

The objective of this task was to generate a computer geometry model of a notional ship that can be used either as a visualization utility or an advanced analysis tool. Moreover, it should be parametric and capable of communicating with external analysis and simulations modules. Overall, it will become a key enabler for a complete dynamic simulation environment of the ship engineering systems, in order to support the analysis of ship operations and studies for survivability and mission effectiveness. The taxonomy of the subsystem components would be predefined and the architecture of the engineering systems would be similar, yet scalable to match the corresponding ship architecture in terms of size and mission requirements.

The original idea that was proposed was to create a baseline notional ship that would be heavily based on the YP-679 Yard Patrol ship configuration. This direction has been proposed by ASDL and encouraged by the feedback advice given by ONR. Yard Patrol (YP) Craft were designed for Midshipmen at the U.S. Naval Academy and are commonly used for training in navigation, ship handling, fleet tactical maneuvering principles, rules of the road, shipboard military procedures, and to gain an appreciation for being at sea.

The YP is of wood hull construction and has an aluminum superstructure. The craft is diesel powered with twin screws and has a range of 1400 nm [1].

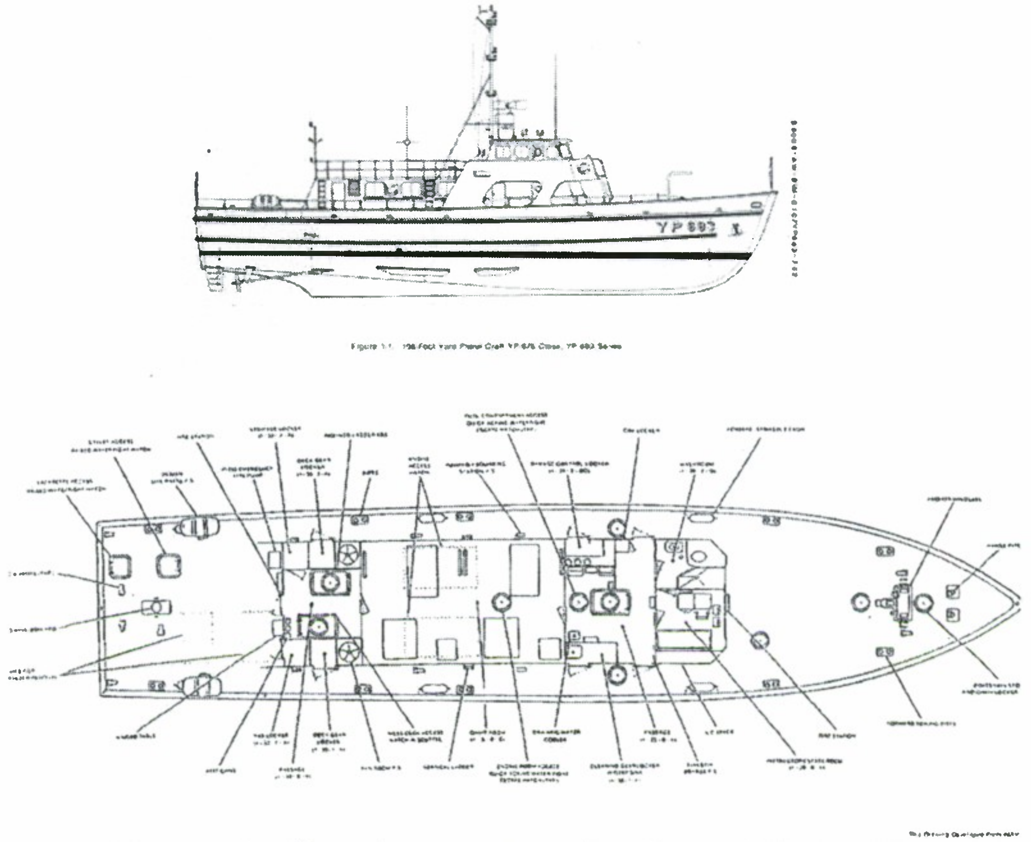


**Figure 6: Profile and top views of a Yard Patrol (YP) ship [1]**

However, despite the compatibility of the available models, the necessary information that became available, either online or directly through ONR was not quite adequate. Thus, the resulting model is notional to great extent, even if the general specifications still adhere well to the actual ship configuration. The only avenue for locating information around the YP geometry could only be found from publicly available resources (web search for reports, schematics and fact sheets). The geometry development was initiated through importing a profile and side view image of the ship (Figure 6), along with basic information provided by US Navy fact sheets [2] . Table 1 below shows the numerical data that has been used for initial sizing of the ship.

**Table 1: General Characteristics for the YP-679 [2]**

| General Characteristics, YP 676 and YP 696 classes | |
|---|---|
| Length Overall | 108 ft. 0 in. |
| Length: Overall: 108 feet (32.9 meters); Waterline Length: 102 feet (31.1 meters). | |

| | |
|---|---|
| Beam, Molded | 22 ft. 9 in. |
| Draft (full load) | 6 ft. |
| Displacement (Full Load) | 173 LT |
| Speed | 12 knots |
| Range | 1800 nautical miles (3300 km). |
| Hull Material | Wood hull, aluminum superstructure |
| Crew: | Officers: 2 Enlisted: 4; |
| **SPECIFICATIONS** | |
| MACHINERY | |
| Main Engine (2) | GM12V-71N, Detroit Diesel 437 BHP each |
| Diesel Generator Sets (2) | 50 kW, 450-Vac, 60 Hz, 3 ph. |
| **CAPACITIES** | |
| Fuel Oil | 6,550 gal. |
| Potable Water | 1530 gal. |
| Lube Oil | 40 gal. |
| Provision Storage | 200 Cu. Ft. Total |
| Cooler | 40 Cu. Ft. Total |
| Freezer | 34 Cu. Ft. Total |

### ***Hull generation and external geometry***

The above images were literally imported into the Paramarine environment and were used as templates for developing the ship geometry. Data listed in Table 1 were also used for initial sizing. Notional information has been added where required information cannot be available.

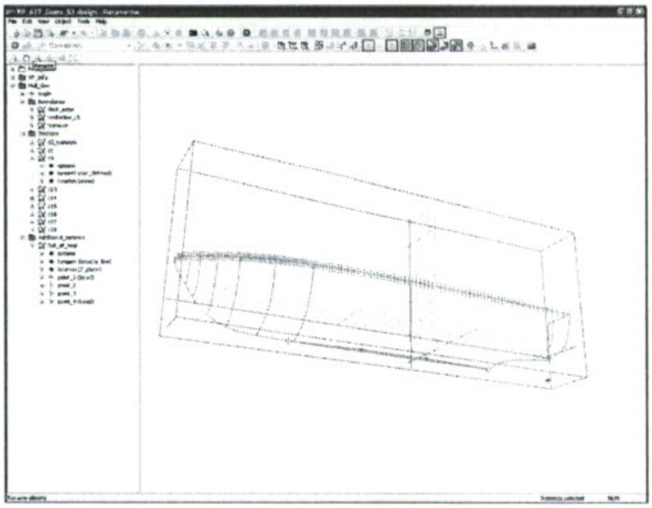


**Figure 7: Early Stages of a Yard Patrol YP-679 Hull Generation in Paramarine**

Figure 7 shows a screenshot of the early stages of hull generation. The virtual "shipbuilding" process is not much different than the actual ship design process or even the manufacturing of the ship. In this case however, except from the centerline and the deck edge curve, all other lines were notional and had been adjusted to the available visual information. Other external elements such as the superstructure, the shaft, the propellers and rudders were sized qualitatively and closed the loop on finalizing the external ship geometry design. The final result is shown in Figure 8, including the hull, superstructure and the bridge, the size and location of the two main diesel engines and components of the propulsion system.

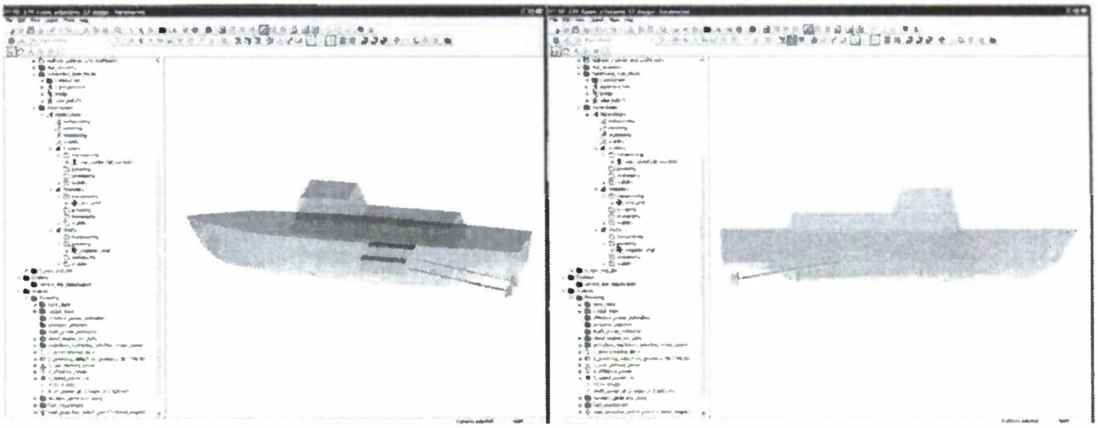


**Figure 8: Final External Design of a Yard Patrol YP-679 in Paramarine**

***Engineering Systems and internal layout***

After completing the geometry for the ship's hull, superstructure, bridge and the internal compartmentation, the next step for completing the overall design would be to include the engineering systems, such as the components of the power generation and distribution system, a cooling system, propulsion, etc. According to the latest technical feedback from ONR, the version of the YP ship that ONR will be using for their own in-house studies, will include an internal systems architecture based off the Tabletop systems simulator. On the other hand, ASDL's own version will use an alternative architecture, which is based on a reduced and scaled down version of the RSAD cooling systems simulation and an in-house developed power system model. For the version that ASDL is preparing for ONR, it is actually convenient that the Tabletop 2.0 simulator is based upon a systems architecture that is greatly similar to the one of an actual YP. Moreover, extensive web search have revealed that the electrical system of a real YP contains about the same number of loads and the cooling system of the Tabletop 2.0 can be easily adjusted and sized to it. A schematic of the YP electrical distribution system is displayed below in Figure 9 and Table 2 contains the load distribution over the entire architecture.

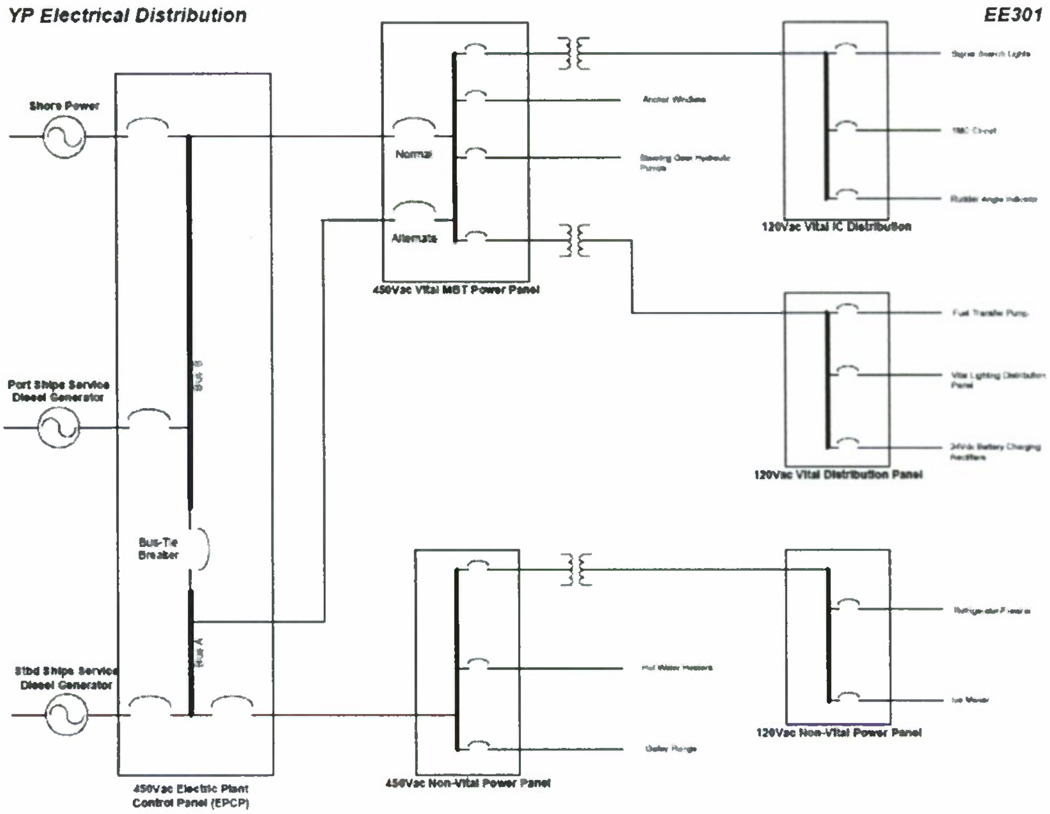

YP Electrical Distribution
EE301
Shore Power
Port Ships Service Diesel Generator
Stbd Ships Service Diesel Generator
Bus B
Bus-Tie Breaker
Bus A
450Vac Electric Plant Control Panel (EPCP)
Normal
Alternate
450Vac Vital MBT Power Panel
Anchor Windlass
Steering Gear Hydraulic Pumps
Signal Search Lights
IMC Circuit
Rudder Angle Indicator
120Vac Vital IC Distribution
Fuel Transfer Pump
Vital Lighting Distribution Panel
24Vdc Battery Charging Rectifiers
120Vac Vital Distribution Panel
Hot Water Heaters
Galley Range
450Vac Non-Vital Power Panel
Refrigerator/Freezer
Ice Maker
120Vac Non-Vital Power Panel

**Figure 9: Electrical power system of a Yard Patrol (YP) Craft [3]**

The Yard Patrol (YP) Craft is equipped with two identical 50-KW, 100A, 450V, 60Hz, 3-phase Ship's Service Diesel Generators (SSDG's). Each SSDG is capable of providing sufficient electrical power for a normal operation of YP's hotel loads [3]. Load clusters are distinguished regarding their importance to the mission and power transfer buses allow for some degree of reconfigurability or resource allocation.

**Table 2: Load list for the YP electrical system [3]**

## List of service loads for a YP ship

**450Vac Vital MBT Power Panel**

Bilge Pump

Engine Room supply Fans with Halon Interlock
Steering Gear Hydraulic Pumps #1, #2
Anchor Windlass

**450Vac Non-Vital Power Panel**
A/C Compressors

**120Vac Non-Vital Distribution Panel**

A/C Cooling Control Power (Messroom and Galley)

Potable Water Pump
Battery Charger Interlock
Battery Hood Blower Fan
Toaster
Refrigerator/Freezer
Dishwasher
Compactor

A/C Seawater Pump
Hot Water Heaters (50gal, 80gal)
Galley Range
Dishwasher Booster Water Heater
Reverse Osmosis Desalinator
Immersion Heaters
Hot Water Circulating Pumps

**120Vac Vital IC Power Panel**
Clear View Screen
Signal Search Lights
1MC Circuit
Radar
Rudder Angle Indicator

**120Vac Vital Distribution Panel**
Fuel Transfer Pump
Sewage System Control
Vital Lighting Distribution
A/C Cooling Unit Control Power
24Vdc Battery Charging Rectifiers

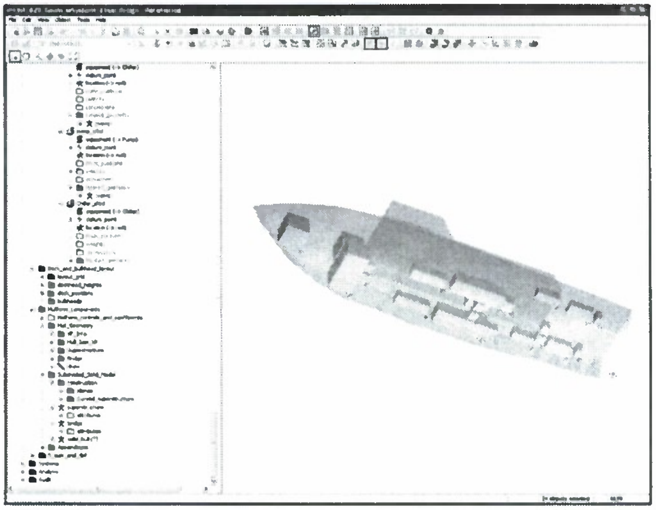
Microwave Oven
Ice Maker
Other Galley Equipment

**24Vdc Emergency Power Distribution Panel**
Panel
Navigation Light Panel
Air Horn Compressor
Underwater Log
Radio Systems
Halon System
E-Call System
Pilothouse Console Dimmer/Lighting
Plotters
Windshield Wipers
Depth Indicator
Emergency Lighting System

## *Complete geometry model*

**At the current phase of this effort, ONR is willing to make use of this model as a visualization environment for its in-house simulation model, which is heavily based on the YP and is also running the Tabletop 2.0 simulation. Thus, is has been instructed that the architecture of the Paramarine model follow the ONR simulator. Figure 10 and**

Figure 11



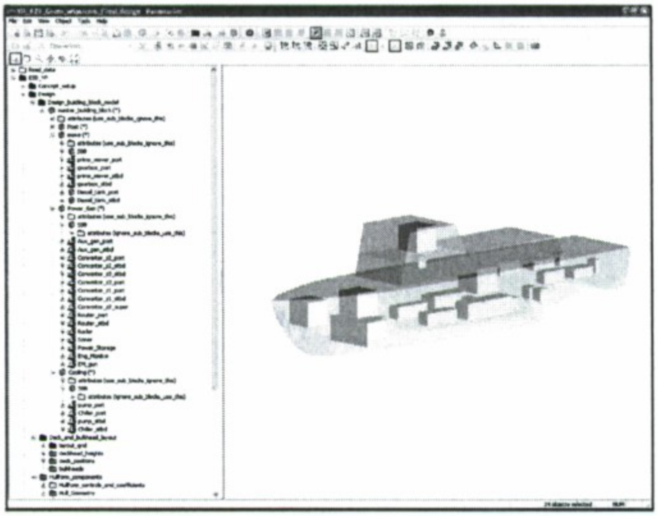


**Figure 10: Final Design of a Yard Patrol YP-679 in Paramarine**

show the final design of the YP, including the subsystem distribution, according to available information. This version of the YP design is subject to future iterations, to further develop the design for matching ONR's needs.

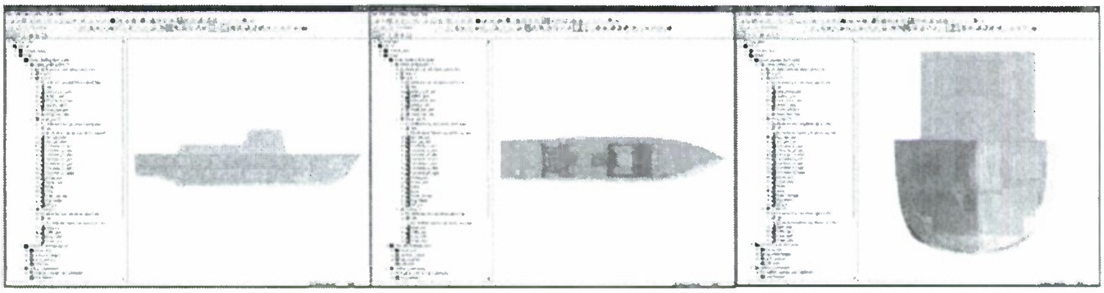


**Figure 11: Three-view images of the YP-679**

The list of subsystems includes the following components:

- Two main diesel engines for propulsion
- Two electric power generators for ship service support
- Seven electric power converters (PCMs) and two routers
- Service loads, representing the ship's radar, sonar and gun
- Two clusters that contain chillers and pumps
- Main control deck
- Battery storage for electric power

For the current model, the above components are defined as objects with properties and performance ratings, in order to allow for ship sizing studies. Typical analyses that can be conducted in Paramarine include weight distribution estimations, stability calculations, cost–effectiveness tradeoffs, etc. The fidelity of the study strongly depends on the modeling detail and this is something up to the user to decide based on his specific needs. Paramarine allows for more elaborate designs that may also include networks for power distribution and cooling, tank and container definitions, detailed structural elements such as bulkheads, inner coating, insulators, etc.

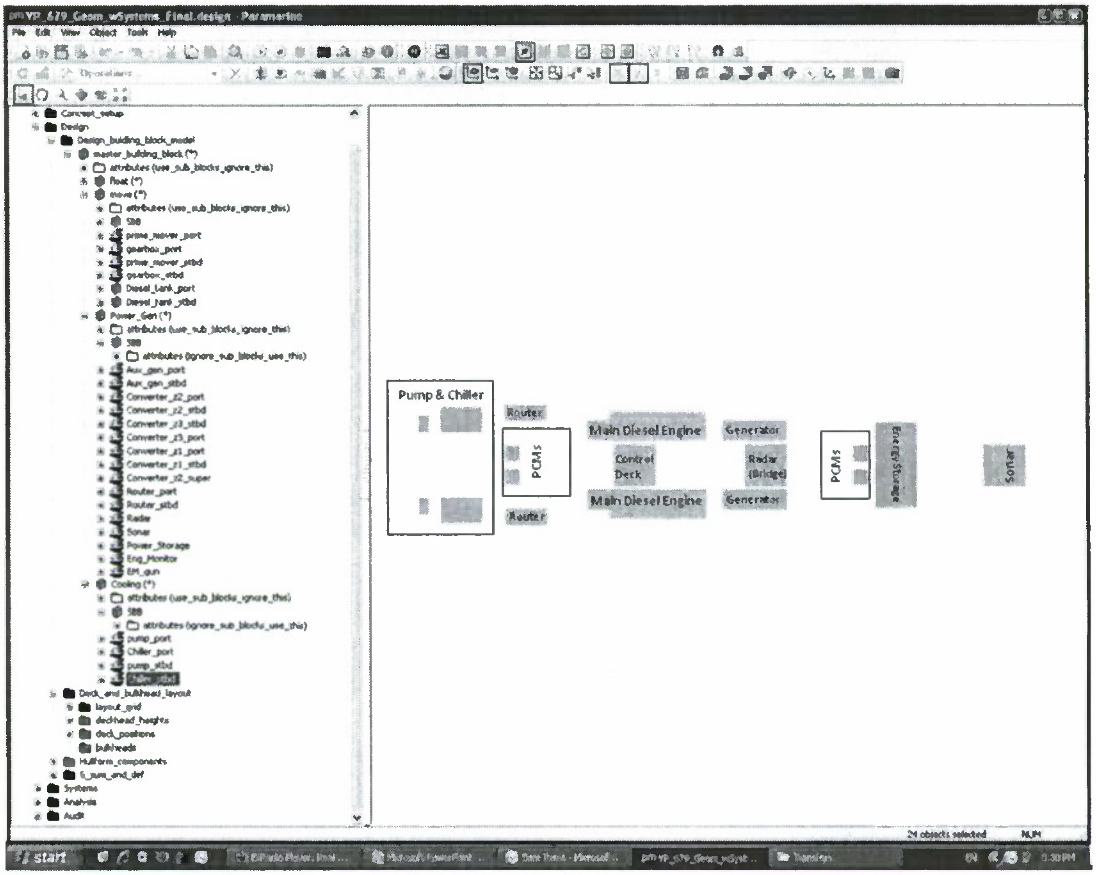


**Figure 12: Distribution of subsystems of the electrical and cooling systems**

## Future Work

For the purpose of using a version of the YP Paramarine model as a visualization environment that is mapped to ONR's advanced systems simulator, the current instance of the model should be adequate. Further enhancement of the design is scheduled, according to ONR's needs and for supporting the experiments that ASDL has planned for future tasks. Power distribution lines and piping networks are going to be added and a more detailed structure of service loads will replace the current setup. Extensive definition of ship spaces that support mission operations will be included, namely the crew working rooms, fuel, ballast and fresh water tanks and rooms that contain vital machinery. Further detailing will also be added regarding the material usage on the ship's structure, as well as more refined definition of structural elements, such as bulkheads, decks, separators, insulators. Eventually, an enhanced design will allow for more insight on the ship's vulnerability, stability and overall effectiveness and provide an analysis capability that can lead to improved designs.

## *Subtask 1.2: Integration of Heterogeneous Dynamic Systems*

### Introduction

Based on the previously described findings regarding co-simulation of third party proprietary dynamic sub-systems, the current work consists of two main tasks: The notional YP model integration must be finalized, and the further refinement of an algorithm that allows co-simulation execution error bounding.

The notional YP model has been modified in some ways to accommodate for easier error tracking. The current effort is to integrate the final version of the agent based controller model, which will then allow to run and test the model in its final form.

The error bounding algorithm is currently under development. Preliminary runs have been made, however the data needs further scrutiny. Different underlying test cases are investigated.

### Integration of Heterogeneous Dynamic Systems

#### *Co-simulation setup for notional YP simulation*

Several improvements have been made to the simulation setup of the notional YP model. The data exchange between the mid level agent based controller ("ABCtrl" in the model) and the low level controller ("metaVDCSxxxxxxxx" in the model) was modified to allow the low level controller to use current information to set valve states properly and to detect and react to ruptures correctly during the current time step. This issue arose due to the fact that previously, all models were executed in parallel. After execution for a given simulation time step, they were stopped, and their data was exchanged. This created the situation that the low level controller got data that corresponded to the previous simulation time step. The decisions made by the low level controller based on this data was not necessarily representative of the current system state, and often led to false reaction in valve states and rupture identification. This problem was overcome by changing the top level scheduler script in the Modelcenter simulation setup such that at any given simulation time step, it runs all models except the low level controller for the length of one simulation time step, stops the models, exchenges the data, and feeds the current necessary data into the low level controller. Only then is the low level controller model executed, this time with the up to date information for the current simulation time step. This resulted in much improved reaction behavior of the low level controller. Ruptures were identified correctly, and the valve states were detected and set more accurately.

For better tracking of system states and data, and a better visualization, a simple interface was implemented in Microsoft Excel. It shows a notional representation of the fluid network. Each working element within the network has been equipped with output windows that show the desired current system states. For valves, the valve state and valve flow is indicated. Pumps show the pump state (on or off, 1 or 0). Chillers and service loads show the respective temperatures. Additionally, every service load has an extra indicator to represent the service load priority as assigned by the high level controller. The variables displayed inside the indicators are also explained in the graph table, together with the current time step. The data to be displayed can be very easily changed within the Modelcenter top level controller script. The implementation of this graphical interface has helped tremendously to track and correct erroneous system behavior. This interface is a preliminary one, to be substituted by the Human Machine Interface described in another section of this report. The final HMI will allow for sophisticated data visualization and user inputs into the simulation during run time. The co-simulation also has a primitive variable output and visualization within Microsoft Excel. This is set up by using standard Excel graphs to track time data of selected variables during the simulation. The variables and graphs are freely formatable. Recently, this interface had to be reprogrammed due to the fact that the variables were displayed in table columns and the simulation values in table rows. Since Excel 2003 only allows for 255 rows, and several variable vectors have a length of 55, the amount of variables that could be displayed was limited. The recent update changed the format such that the variables are now in rows and the data in columns. Therefore, a total of 65533 variables can be displayed (the first two rows are for the simulation time and iteration number) at a maximum of 254 runs. The new Excel 2007 allows for more outputs due to larger amount of available rows and columns.

### ***Model validation and error bounding***

In a previous report, it was mentioned that algorithms for the numerical solution of differential equations are investigated for their usefulness towards the evaluation of a co-simulation of dynamic systems, as for example in the notional YP. The idea was that the underlying problem is similar: Approximate a solution to a curve whose shape is not known. This however, is not possible. The reason is that a differential equation is firmly defined at each point. The relation between inputs and outputs are fixed by the given equation, and any function evaluation will always give the correct result. In a co-simulation, it is not known whether the result is correct or not, hence the algorithms will not work for such a problem.

However, the idea to apply such numerical analysis algorithms to co-simulation has another aspect, namely that of adjusted time steps. There are various algorithms in numerical analysis that enable to adjust the time step of the simulation according to given tolerances (for example, the Runge-Kutta-Fehlberg [RKF45] and Adams Variable Step-Size Predictor-Corrector methods). While such algorithms naturally are usually used to numerically solve differential equations, the application and underlying problem is similar. If the curve changes quickly, the time step needs to be small in order to catch the changing system states more accurately. If the curve changes slowly, the time step can be larger without losing accuracy. For complex simulations, the real time savings for simulation execution due to such approaches can potentially be significant. However, these algorithms require a significant amount of function calls, each of which represents a complete co-simulation cycle, is computationally expensive and may diminish or even reverse the advantages due to the time step adjustments. This is because in order for these algorithms to achieve their desired accuracy, in a first step they calculate various future points to try to find a slope that better represents the underlying curve. In a second step, the found results are then verified by yet another function call. Due to their nature, such methods are referred to as predictor-corrector methods. While computationally expensive, they allow for high accuracy and the binding of the error to within a given tolerance level. The first step will therefore be to implement such algorithms for co-simulation, by testing them for given sample cases and see whether their application is feasible. In a second step, methods to reduce the amount of function calls will be evaluated, in the hope to find a method to create a surrogate model that can determine optimal time steps for given error tolerances without the computational expenses of the original algorithms. The surrogate model would represent a function that maps the time step obtained from the expensive algorithms at a given tolerance level to a simpler variable that can be calculated easier, such as the second derivative. Ideally, this would be a universal map that would not depend on any parameter other than the one chosen to be the mapping variable. Also, unlike the original algorithms that depend on the magnitude of the current values (this is due to the fact that an ODE has the form y' = f(t, y), thus being a function of the current state of the variable), it may turn out that the magnitude is not a defining factor in such a mapping. This can be explained by looking at the simple case of the second derivative as the mapping variable, as explained below: The second derivative is only a function of the difference between the past, current, and future value of the variable, but not the magnitude of it. Hence, the absolute magnitude would not be a factor, only the development and dynamic behavior of the curve itself.

The basic idea is to use the second derivative of a curve as an indicator of how the step size should be varied. If the second derivative of a curve is low, its slope changes slowly, and the step size can be large. If it is high, the slope changes quickly, and the step size should be low to accommodate for the rapid change. A preliminary investigation into this approach was done in a recent paper (Nairouz, B., Hoepfer, M.: "Investigations for Time Step Settings in a Dynamic System Co-Simulation Environment", Electric Ship Design Symposium, Washington D.C., Jan 2009). However, the method presented in this paper was a first approach, and more of a proof of concept to show that the second derivative of a curve can be used to adapt the simulation time step. All variables and parameters there were chosen arbitrarily. Thus, the results were not optimal, and there was no error bound. As described above, the proposed algorithm would significantly reduce the amount of function evaluations necessary, but still deliver a time step that represents the necessary minimum time step for a given tolerance. As mentioned, ideally this would be a universal algorithm that does not depend on the dynamic behavior of the underlying systems, but only on the current state of the variable or variables under observation. Since in a co-simulation, there are multiple variables to be examined, the algorithm will likely have to be applied to each dynamic variable that is exchanged between sub-models. This will even more increase the value of such an algorithm, since it will help to reduce the co-simulation execution time. The time step chosen for the co-simulation environment would have to be the minimum time step of all the variables under consideration.

The programming of such methods is currently ongoing. They will be tested first with given functions and their known behavior, to test the general applicability of the approach.

It must also be ensured that the magnitude of the variable is not a factor, and that a correct mapping variable is chosen. A first implementation of the runge-Kutta-Fehlberg algorithm is working properly, and a preliminary map of the time steps onto the second derivative has been performed. The data is as of now not refined, since the algorithm has shown to be offset by one time step compared to the mapping variable. Further, the calculation of the second derivative from variable states is not yet refined enough to represent the change in slope accurately enough. Hence, the algorithm needs further refinement. After it is established that the algorithms can be employed for such problems, they will be tested with a simple dynamic model which consists of two sub models. The monolithic model behavior for this simple co-simulation is known, and hence the algorithm output can be compared with the "true" system behavior to see how well this implementation works. A more complicated model exists in the ASDL lab, and will be used for further studies on the subject once the initial implementation and testing has been successful. The long term goal is to find a universal mapping of time step setting to model behavior for any given co-simulation where the state variable magnitudes are know at each time step. In a final step, this method, once proven, would have to be implemented into the notional YP model, to prove its applicability to a real world problem.

### Future Work

The current co-simulation setup for the notional YP is not yet fully functional, and further work will be needed to get the model to run correctly. As of currently, the model is being upgraded by the final version of the mid level agent based controller, with the final inference engine installed. This has to this date shown to cause some problems with the setup of the current computer system. Several software components needed to be reinstalled or upgraded. This is currently ongoing. Once the underlying software issues are resolved, the models can be implemented into the Modelcenter environment, and "wired" together within the top level control script. The developed visual interfaces will help in this attempt, but finally the developed HMI will be integrated and allow for better data visualization, data storage, "replay" of simulation runs, and active user inputs during simulation execution. The approach for using and modifying numerical integration schemes and algorithms for ordinary differential equations will be continued. Existing algorithms will be implemented such that they accommodate for the new application in co-simulation. Various algorithms will be considered, as described above. Their applicability and usability will be further tested. Basic functions with known behavior, and later a simple dynamic model will be used for this. If proven successful, a more complicated model will be used as well, which introduces stiffness and nonlinearities, and thus provides a more challenging basis for feasibility studies.

# Task 2: Intelligent Autonomous System

## *Subtask 2.1: High Level Control*

### Introduction

IRIS designed systems are complex in nature and the system consists of multiple subsystems which provide necessary functionalities to the system. Thus, in the operation of such complex systems, the control systems are hierarchical in nature. The controllers at each level has their own objectives and they collaborate together to achieve the overall operation goals. The effectiveness of an intelligent system depends on the functionalities that the controllers at different levels provide, which results from the objective decomposition and the use of control techniques at each control level. In the resource allocation process, since the resources are limited, a well designed control systems are needed to make right decisions and control commands in order to efficiently utilize the resources. IRIS control systems consist of three levels of controls: high-level control, mid-level control and low-level control. Each control level has different control objectives and employs a varied method or technique to fulfill its functionalities.

In the IRIS control architecture, the high level control is responsible for setting policies regarding all the subsystems. Currently, it accomplishes this task by prioritizing service loads, i.e. deciding what is important at which point in time, and how important in comparison to others. This prioritization is based on the evaluation of operating environment, ship status and mission being performed.

## Development of High Level Control for Resource Allocation

Several promising approaches were investigated for high level control development, including partially observable Markov decision process, reinforcement learning and rule based control. In addition, an approach that uses an optimization function that takes into account relevant system parameters and mission status was also investigated. It was found that a rule based controller is effective in making autonomous decision and easy to be implemented when allocating resources to service loads. This is due to the fact that high level decisions about the system priorities are made by the controller, which is effectively to be realized by a rule based reasoning process. In addition, the rule based controller is easy to be implemented and developed for this purpose, and is flexible in modification and extension for dealing with multiple-resource allocation problem.

The rule-based high level controller possesses a certain degree of intelligence when deciding on the load prioritization. It makes decisions guiding the resource allocation based on the mission that the ship is performing and the resource status a service load has. It determines the priority in which a service load get resource by evaluating how a service load is important to best perform the current mission based on the situation at hand. It also depends on if a service load requires resource urgently. For example, in this study the high level control is applied to reallocate the cooling resource by prioritizing the ship systems. This prioritization is determined by the evaluation of environmental state, system status and mission being performed. The developed high level controller possesses intelligence when deciding on the prioritization for the ships systems. The priority not only depends on the importance of a system to the overall mission, but also depends on its urgency to require the resource. The output from the high level controller is the percentage of maximum flow rate that a system can get based on the evaluation of its importance to the mission and urgency to get resource.

The decision rule applied in the high level controller indicates that if a system has high importance because it is critical to the current mission, but this system has a large margin before it reaches a critical stage (e.g. not extremely hot but running at a regular operating temperature), then it makes sense to save some of the resources and not provide this load with further resource. This formulation implemented in the high level controller has been proved to increase the efficiency of resource usage and mission effectiveness. The high level controller will give this percentage value to the mid and low level controllers which will control the corresponding valves and find an optimal route to distribute the required resource to the ship systems.

### Future Work

Since there are different type of resources on shipboard need to be effectively reallocated. Further work related to the high level control is to address the allocation of some non-recyclable resource such as electrical power. This will require a different formulation on the high level decision making rule and algorithm. In addition, it is necessary to address the interactions between resources when allocating the resources.

## *Subtask 2.2: Multi-Agent Based Mid-level Control with Dynamic Inference Engine*

### Introduction to Distributed Dynamic Probabilistic Inference Engine

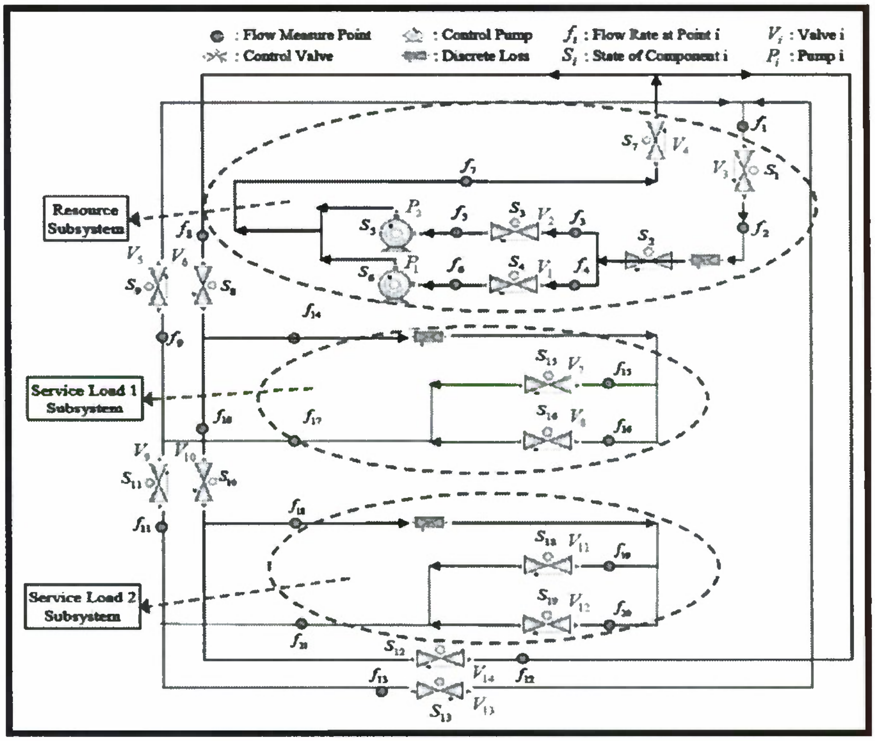
Identification of the state of a system over a span of time is generally called monitoring or filtering. For a dynamic system with uncertainty, the monitoring or filtering process is stochastic and the goal of inference is to maintain a probability distribution over the state of the system at each time point, based on the evidences/observations available up to that point. There are a few reasons for a large scale complex system to use a distributed dynamic inference engine for state estimations: incomplete and uncertain data set, insufficient system knowledge, distributed subsystems, and limited computation and communication capabilities of local processors. A distributed inference engine can be modeled as a multi-agent system. Each agent represents certain local knowledge of subsystems. There are two main challenges in applying distributed inference engines: global coordination among distributed nodes and robustness to partial damages.

Extended from junction tree belief propagation in individual Bayesian networks, Multiple Sectioned Bayesian Networks (MSBNs) as probabilistic distributed inference engine were first proposed by Xiang et al. to give the principles of MSBNs and its corresponding constraints and implementations [18]. MSBNs support general large scale complex systems. Communication efficiency of MSBNs is improved through breaking shared variables between sub Bayesian networks into smaller linkages. In addition, individual sub Bayesian network agent can keep privacies from other sub Bayesian networks and a set of multiple queries can be made simultaneously in MSBNs.

Multi-agent based control with distributed dynamic MSBNs had been established for a simplified fluid network model with one chilled-pump unit and two service loads as sketched in Figure 13. Detailed result analysis had been discussed in the previous reports. In the past few months, the simplified fluid network model has been extended to a notional YP chilled water system, which includes two chiller-pump units and 7 service loads. Multi-agent based control with distributed dynamic MSBNs has been implemented into the extended fluid network. A nominal case has been tested and the results have been analyzed.

### Implementation of Distributed Dynamic Probabilistic Inference Engine for Ship Chilled Water System of the Notional YP Model

The sketch of the chilled water system of notional YP with two chiller-pump units and 7 service loads is shown in Figure 14 and the possible observable flow rate nodes distributed in the fluid network are shown in Figure 15. The entire Bayesian network for the whole system is shown in Figure 16.



● : Flow Measure Point
: Control Valve
: Control Pump
: Discrete Loss
$f_i$ : Flow Rate at Point i
$S_i$ : State of Component i
$V_i$ : Valve i
$P_i$ : Pump i
Resource Subsystem
Service Load 1 Subsystem
Service Load 2 Subsystem
$f_1$ $f_2$ $f_3$ $f_4$ $f_5$ $f_6$ $f_7$ $f_8$ $f_9$ $f_{11}$ $f_{14}$ $f_{15}$ $f_{16}$ $f_{17}$ $f_{18}$ $f_{19}$ $f_{20}$ $f_{21}$ $f_{22}$ $f_{23}$
$S_1$ $S_2$ $S_3$ $S_4$ $S_5$ $S_6$ $S_7$ $S_8$ $S_9$ $S_{10}$ $S_{11}$ $S_{12}$ $S_{13}$ $S_{14}$ $S_{15}$ $S_{18}$ $S_{19}$
$V_1$ $V_2$ $V_3$ $V_4$ $V_5$ $V_6$ $V_7$ $V_8$ $V_9$ $V_{10}$ $V_{11}$ $V_{12}$ $V_{13}$ $V_{14}$
$P_1$ $P_2$

**Figure 13: The Simplified Chilled Water System Sketch**

From Figure 16, we can see that the structure of the monolithic Bayesian network is too complex and cumbersome for the notional YP chilled water system. There are four main reasons to establish a distributed probabilistic inference engine for the notional YP chilled water system:

- The fluid network is highly distributed over the entire ship physically.
- Computation intensity for a monolith Bayesian of a complex system is high.
- A monolithic Bayesian network suffers from a single point failure and the system reliability is reduced.
- Communication cost is high by gathering the distributed environment information to a center processor.

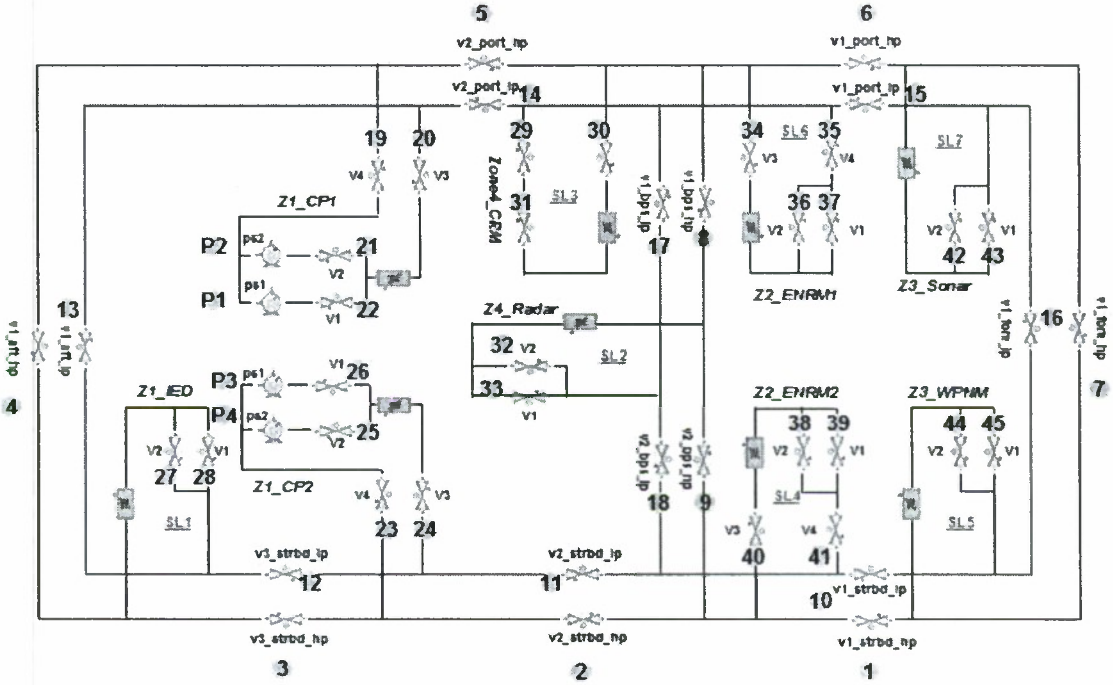


Figure 14: The Notional YP Chilled Water System Sketch

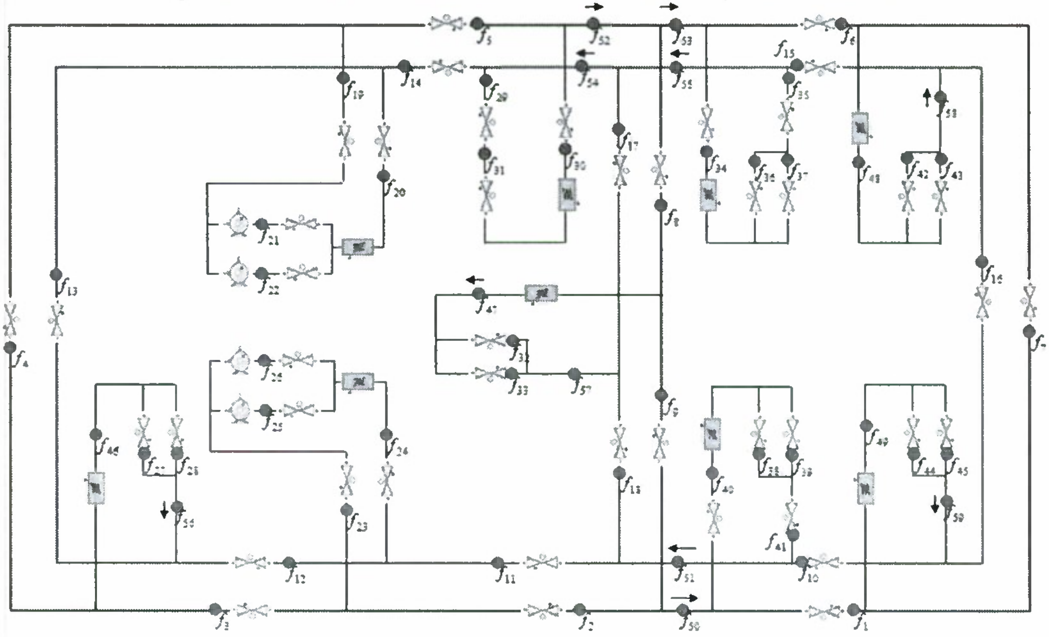


Figure 15: Possible Observable Flow Rate Node Distribution in the Notional YP Chilled Water System

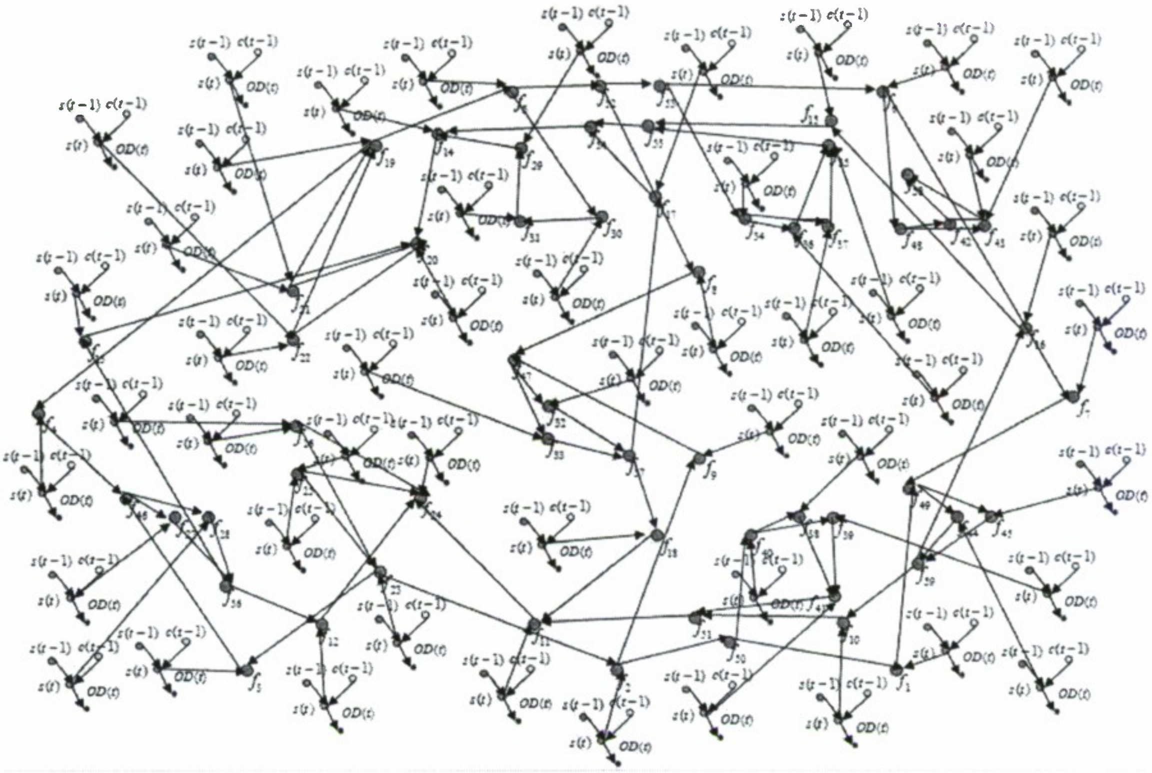


**Figure 16: Bayesian Network for the Whole Notional YP Chilled Water System**

By using MSBNs, the entire notional YP chilled water system is decomposed into 9 different sub systems. Each sub system has a corresponding sub Bayesian network. The decomposition and connections among the sub system Bayesian networks are shown in Figure 17. The structures of chiller-pump unit 1, chiller-pump unit 2 and service load 1 sub Bayesian networks as three typical sub Bayesian networks are shown in Figure 18 and Figure 19. As we can see, a two time-slice homogeneous Dynamic Bayesian Network (DBN) is used for each component state evolution from time t to time t+1. The fully factorized Boyen-Koller (BK) approximation algorithm is used for each local sub Bayesian network belief updating over the time span and the static Junction Forest Linkage Tree (JFLT) algorithm is used for the global system belief updating over the entire network.

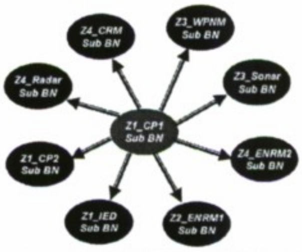


**Figure 17: Bayesian Network Decomposition and Connection for the Whole Notional YP Chilled Water System**

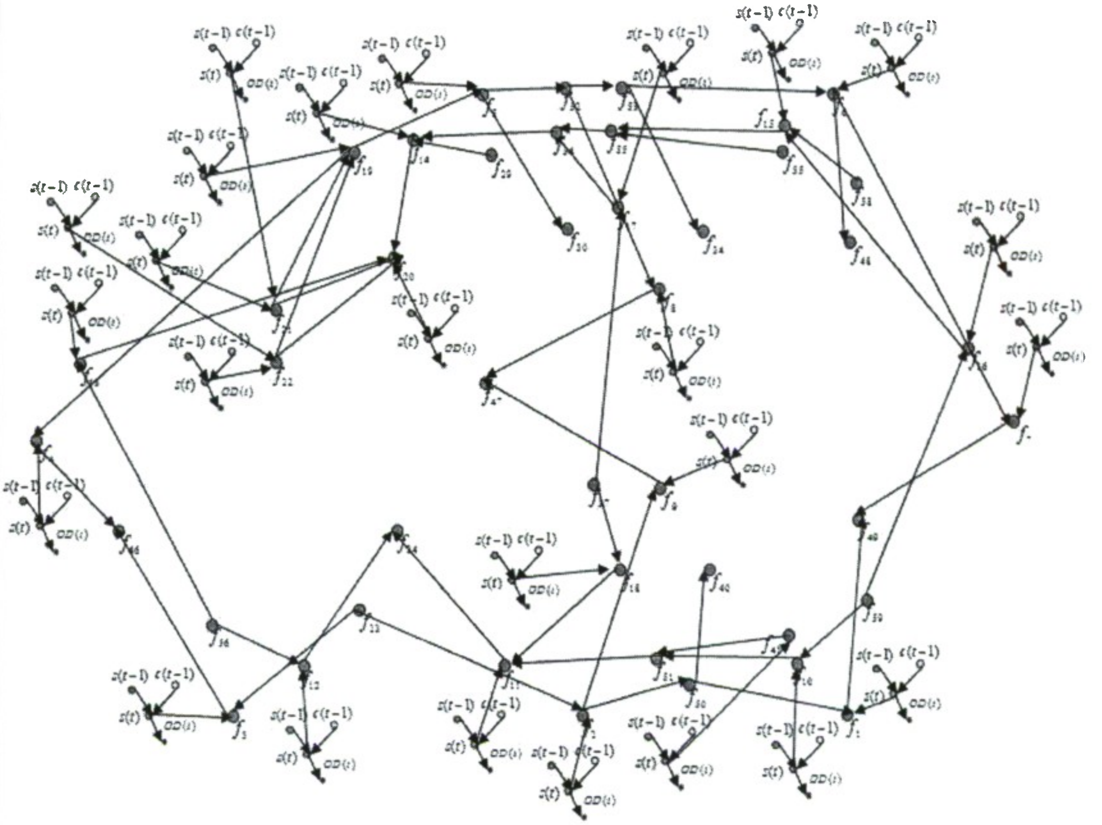


**Figure 18: Sub Bayesian Network of Chiller-Pump Unit 1 of the Notional YP chilled Water System**

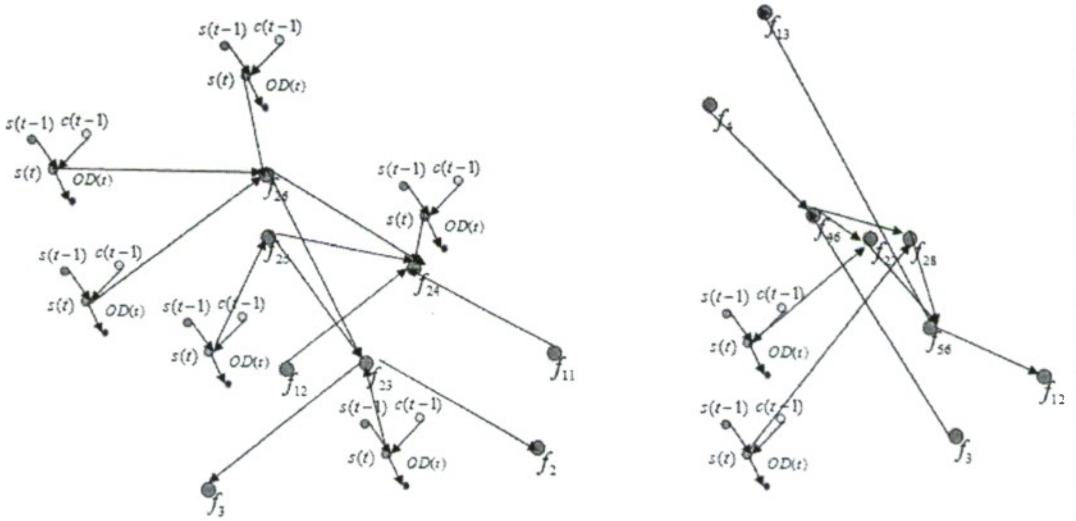


**Figure 19: Sub Bayesian Networks of Chiller-Pump Unit 2 and Service Load 1 of the Notional YP chilled Water System**

The simulation environment established in ModelCenter shown in Figure 20 is as the same as described in the previous reports. Therefore, the detailed discussion is neglected in this report. A nominal case (no damage and the complete set of sensors) is tested by running the whole simulation for 120 seconds.

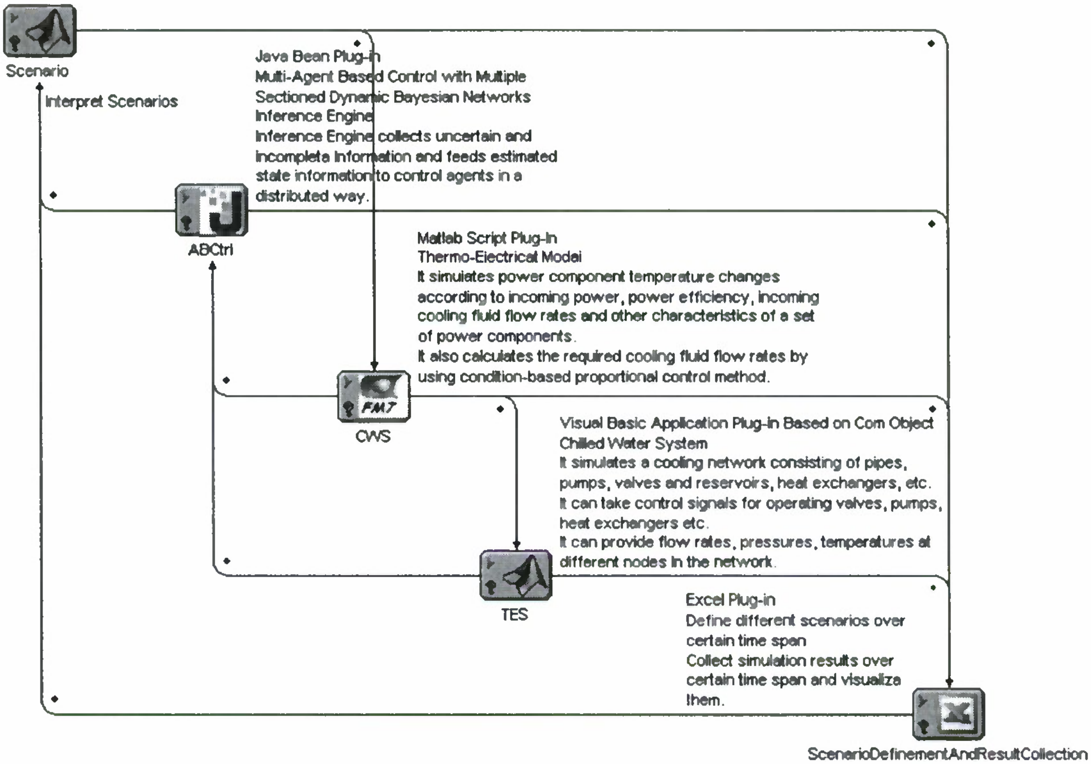

Scenario
Interpret Scenarios
Java Bean Plug-in
Multi-Agent Based Control with Multiple
Sectioned Dynamic Bayesian Networks
Inference Engine
Inference Engine collects uncertain and
incomplete information and feeds estimated
state information to control agents in a
distributed way.
ABCtrl
Matlab Script Plug-In
Thermo-Electrical Model
It simulates power component temperature changes
according to incoming power, power efficiency, incoming
cooling fluid flow rates and other characteristics of a set
of power components.
It also calculates the required cooling fluid flow rates by
using condition-based proportional control method.
FM7
CWS
Visual Basic Application Plug-In Based on Com Object
Chilled Water System
It simulates a cooling network consisting of pipes,
pumps, valves and reservoirs, heat exchangers, etc.
It can take control signals for operating valves, pumps,
heat exchangers etc.
It can provide flow rates, pressures, temperatures at
different nodes in the network.
TES
Excel Plug-in
Define different scenarios over
certain time span
Collect simulation results over
certain time span and visualize
them.
ScenarioDefinementAndResultCollection

**Figure 20: The Whole Test Environment Analysis View in ModelCenter**

## Result Analysis (Nominal Case)

Conditions: all of the components are not damaged; every flow rate point listed in the Bayesian network is observable; every component open degree is observable; both of the resource capacities of pump-chiller unit 1 and pump-chiller unit 2 are 0.8*kg/sec*; the initial temperatures of service load 1 to service load 7 are 370K, 360K, 350K, 340K, 283K, 293K and 303K respectively. The monitored outputs are shown in from Figure 21 to Figure 28.

Initially, service load 1, service load 2, service load 3, service load 4 and service load 7 temperatures are above the temperature threshold, while service load 5 and service load 6 temperatures are below the threshold. Service load 1, service load 2, service load 3, service load 4 and service load 7 require cooling water. The summation of their requirements is above either of pump-chiller unit 1 or pump-chiller unit 2's resource capacity, therefore, both of the two pump-chiller units are operating to provide enough cooling water to the necessary service loads. Since service load 5 and service load 6 as two power components have 100*kw* incoming power and their efficiencies are 0.7, 30 percent of the incoming power dissipates into service load 1 and service load 2 and causes their temperatures to increase. All of the service load temperature changes over time are shown in Figure 21 and we can see the service loads with higher initial temperatures are cooling down while the service loads with lower initial temperatures are heating up and they are converging to some stable conditions. From Figure 21 to Figure 28, we can see actual flow rate of each service load does follow the trend of its corresponding flow rate requirement with one time step delay. However, the desired flow rate and the actual flow rate for each service load do not match exactly all of the time. Currently, the low level controller for adjusting the valve open degree to satisfy the flow rate requirement for each service load is a simple proportional feedback control. Considering the fluid network model running in Flowmaster is slow, the low level feed back control only runs 4 iterations for each new desired flow rate to speed up the whole running process. Increasing the number of iterations between the low level controller and the fluid network model will make the actual flow rate follow its corresponding desired flow rate of each service load more closely. In summary, for the nominal case, the control system with MSBNs inference engine can make the right decisions and distribute the resource to different service loads accordingly.

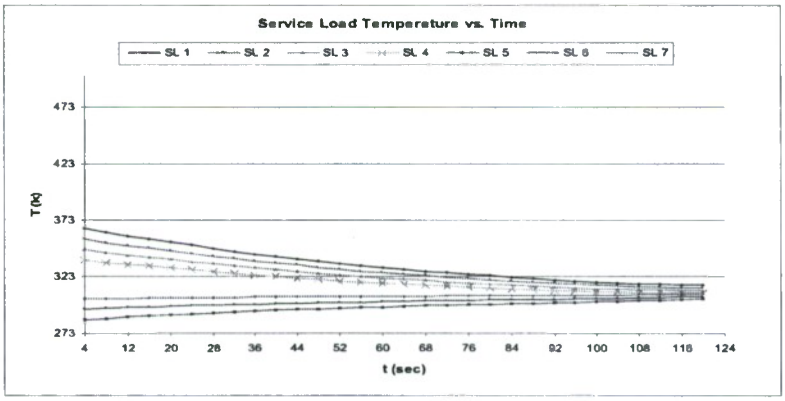


**Figure 21: Nominal Conditions: Service Load Temperature vs. Time**

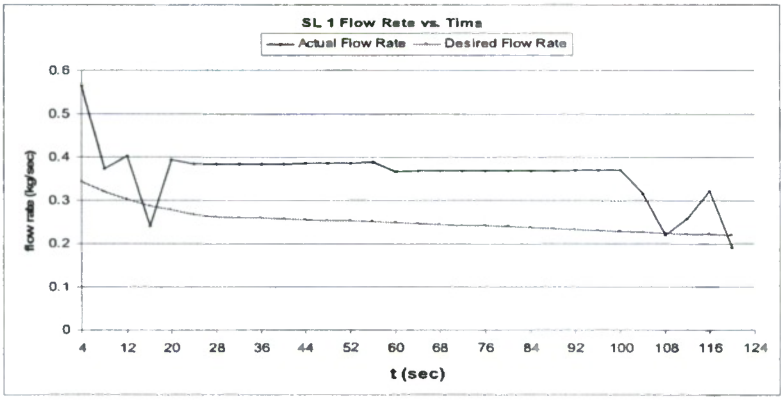


**Figure 22: Nominal Conditions: Service Load 1 Flow Rate vs. Time**

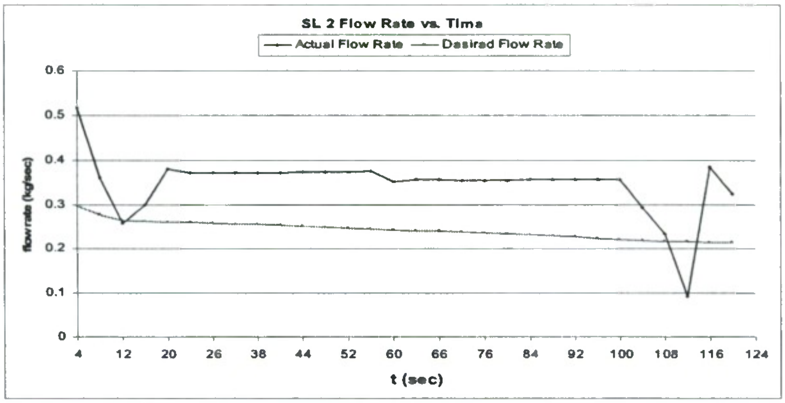


**Figure 23: Nominal Conditions: Service Load 2 Flow Rate vs. Time**

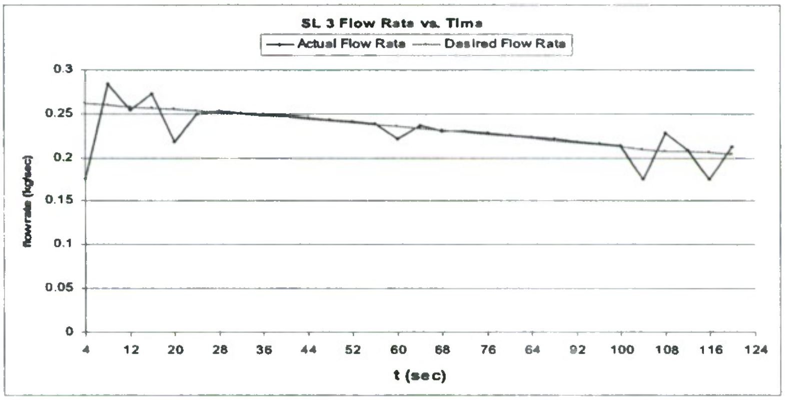


**Figure 24: Nominal Conditions: Service Load 3 Flow Rate vs. Time**

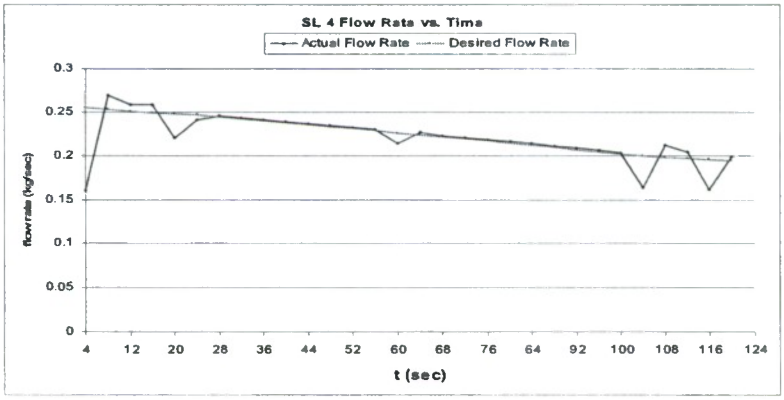


**Figure 25: Nominal Conditions: Service Load 4 Flow Rate vs. Time**

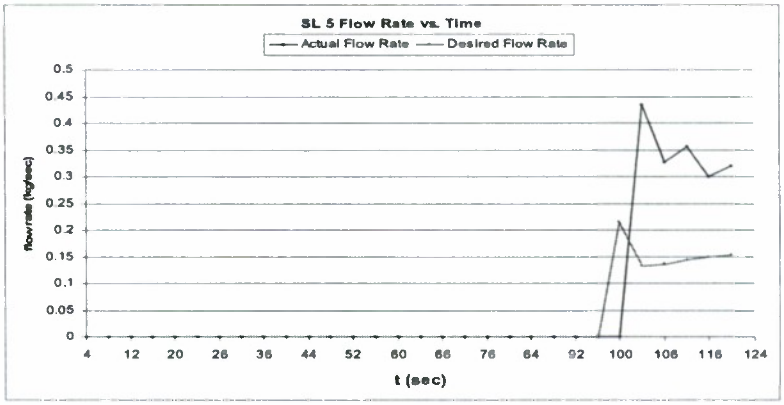


Figure 26: Nominal Conditions: Service Load 5 Flow Rate vs. Time

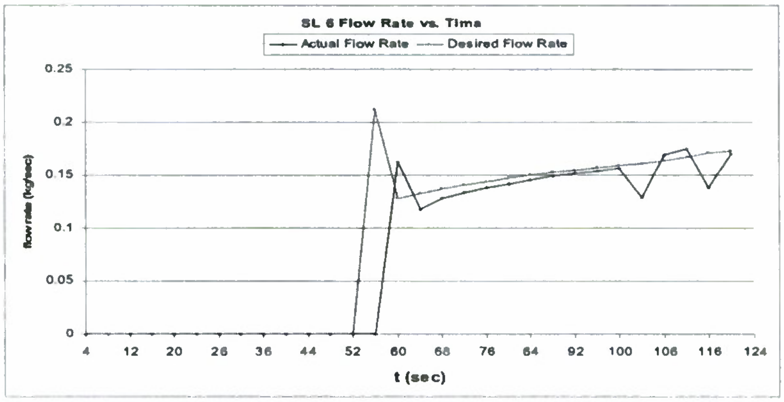


Figure 27: Nominal Conditions: Service Load 6 Flow Rate vs. Time

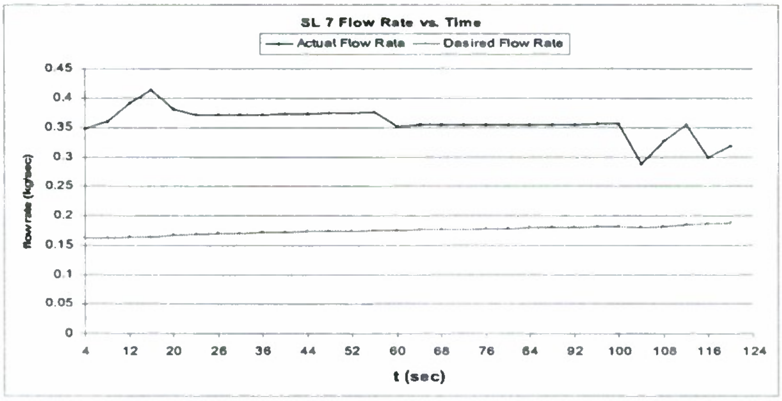


Figure 28: Nominal Conditions: Service Load 7 Flow Rate vs. Time

### Future Work

In the current implementation, sensor types, sensor quantities, and sensor locations are fixed. The inference engine is using the available information to perform state estimations. In many practical systems, sensors are added in an ad hoc manner by the engineers using their experiences in a qualitative way. Quantitative evaluation of sensor cost and performance will help design a sensor system which maintains certain performance level at the lowest cost. A sensor system design includes many factors, such as sensor types, sensor quantities, sensor locations, sensor cost, inference accuracy, etc. There is no straightforward relation between the number of sensors and inference accuracy. The relevance of information gathered by different sensors has significant impact on inference accuracy. In summary, a sensor network optimization for on-line automatic control of a large-scale complex system includes four aspects:

- Inference accuracy: determining state estimation accuracy of certain components, subsystems or the whole system.
- Inference efficiency: determining how fast the information can be handled to make inference for certain components, subsystems or the whole system.
- Minimal sensor set: finding a minimal sensor set which achieve a specific degree of inference accuracy of certain components, subsystems or the whole system.
- Minimal cost sensors: in the case that different sensors are assigned with different costs, finding the minimal cost of a sensor set which can achieve a specific degree of inference accuracy of certain components, subsystems or the whole system.

For a distributed Bayesian network as described in this report, data quality and availability can be simulated by observability of nodes. Through simulation and optimization methods, different sensor set and sensor deployment could be evaluated and the optimized one could be obtained.

## Task 3: Graph-Based Component Surrogate Modeling

### Introduction

The goal of this task is to develop an M&S method for a chilled-water network that is capable of taking the connection-topological configuration among the components of the network as a "design variable." Then with the integrated the design-space exploration or the design optimization process, this proposed M&S environment may enable a simulation-based design for resiliency and survivability. The method consists of three key ideas – a graph-based topological modeling, object-oriented component model definition, and surrogate modeling for representing components' behaviors. Though the development of the method is based on the application for fluid systems modeling, the development approach has also considered more extended uses including the application to an electric power network which is another most common type of physics networks in engineering systems, just with minimal modifications of the method.

### Progress Summary

The task has been managed by being separated to five phases, which is:

Phase 1: Development and test of the basic classes for a simulation environment like a solver and a data manager and the classes of the graph elements such as nodes, edges, sources, sinks, and damages.

Phase 2: Development of the damage scenario generator class. Test of the damage simulation of the M&S environment

Phase 3: Development and test of a reference damage controller.

Phase 4: Development and test of the experimental design class for the network topological space.

Phase 5: Building the model of the chilled-water system of the notional YP. Demonstration of damage analyses and topological design space exploration.

So far, Phases 1 to 3 were all completed. For Phase 4, the task was modified and about 90% of the task has been finished. For Phase 5, both building the surrogate model of the notional YP fluid model and damage analysis demonstration were completed, but the plan of topological design space exploration was changed to the optimal damage control valve placement problem. Due to this change of the demonstration plan, the task of Phase 4 was also changed to the development of the design exploration environment for the smart valve placement

## Damage Analysis Demonstration

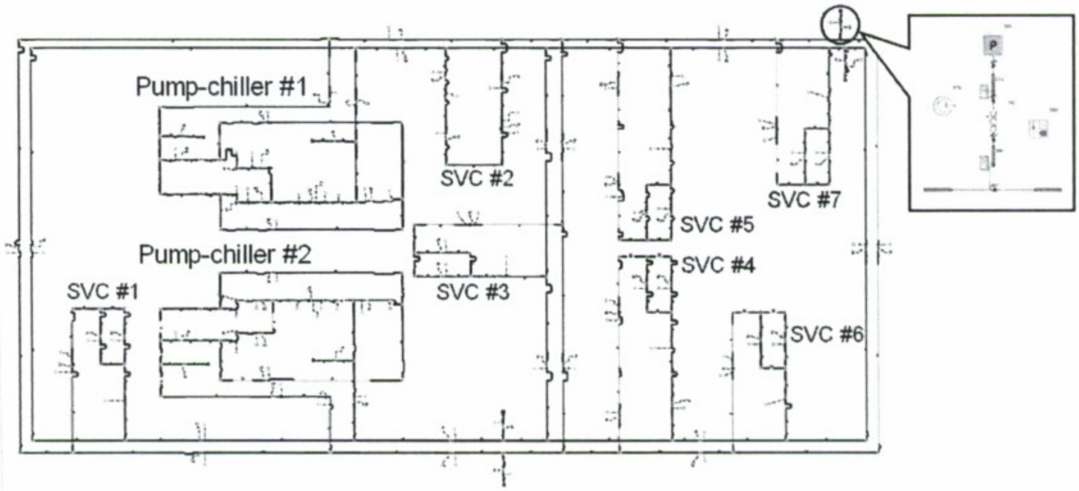


**Figure 29: Notional YP Fluid Model**

Figure 29 is the Flowmaster V7 model of the notional YP fluid system. As a summary, the notional YP fluid model consists of a pair of redundant pump-chiller sub-networks and seven heat exchanger units for serving six thermal service loads. This is because the heat exchanger units SVC no. 4 and SVC no. 5 are a redundant pair of serving a single thermal load. Each heat exchanger has either one or two (as redundancy) flow control valves in it. The system has 18 damage control valves, 11 flow control valves, and 10 valves for load isolation, but in the damage analysis, only the damage control valves were connected to the control model for simulation.

Figure 29 also describes how damage modeling was implemented in this model. Applying the same approach that the Navy M&S used, a rupture valve, which is shown on the upper right corner of Figure 29, was created at a location on which a damage was scheduled to occur during a simulation.

## Graph-based surrogate modeling

The first step was the translation of the notional-YP fluid model into a graph-based representation, such as the one given in Figure 30. In this representation, the model had 52 edge components and 40 nodes, but all the components belonged to one of five different component types. In other words, all the edge components could be represented by five component models with appropriate choices of model parameters for higher reusability. Table 3 is the list of the component models defined in the graph-based model representation.

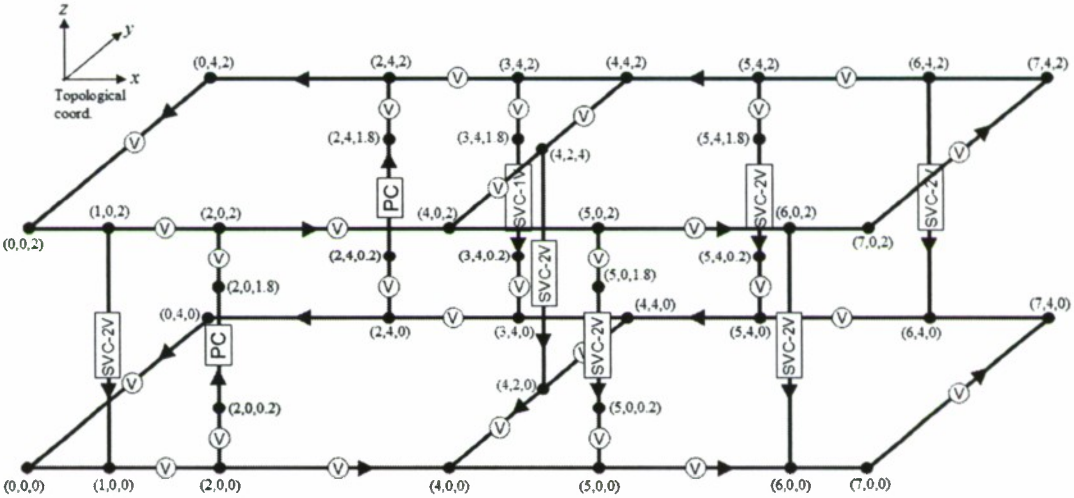


**Figure 30: Graph Representation of Notional YP Fluid Model**

**Table 3: Edge Component Types**

| Name | Component type | No. of comps | State vars | Boundary cond. | Control vars | Parameters |
|---|---|---|---|---|---|---|
| PIPE | Simple pipe | 14 | $q$ ($m^3/s$) | $\Delta P$ ($Pa$) | None | $l$ (length,$m$) |
| VPIPE | Pipe with a valve | 30 | $q$ | $\Delta P$ | $v_1$ (0 to 1) | $l$, $d$ (diameter,$m$) |
| SVC-1V | Svc load with a valve | 1 | $q$ | $\Delta P$ | $v_1$ | None |
| SVC-2V | Svc load with two valves | 6 | $q$ | $\Delta P$ | $v_1, v_2$ | None |
| PC | Pump-chiller sub-net | 2 | $q_{in}, q_{out}$ | $P_{in}, P_{out}$ | $v_1, \omega_{pmp}$ (rad/s) | None |

## Generation of component surrogate models

**Eight neural-net (NN) surrogate models were created to model the five different types of components described in Table 3.**

Table 4 shows the specification of the NN-surrogate models.

**Table 4: NN-Surrogate Model Specifications**

| Name | NN-functions | Input range | | Data size | No. of nodes | Training MSE | Test MSE |
|---|---|---|---|---|---|---|---|
| PIPE | pipe | $-0.001 \le q \le 0.001$<br>$-10^4 \le \Delta P \le 10^4$<br>$0.5 \le l \le 5$ | | 21,511 | 5 | $1.1351 \times 10^{-4}$ | $3.1619 \times 10^{-4}$ |
| VPIPE | vpipe_0127 | $d = 0.0127$<br>$0.3 \le l \le 1.5$ | $-0.001 \le q \le 0.001$<br>$-10^5 \le \Delta P \le 10^5$<br>$0 \le v_1 \le 1$ | 37,817 | 7 | $8.3003 \times 10^{-4}$ | $1.7213 \times 10^{-3}$ |
| | vpipe_01905 | $d = 0.01905$<br>$0.3 \le l \le 6.5$ | | 102,285 | 7 | $1.9484 \times 10^{-3}$ | $2.9164 \times 10^{-3}$ |
| | vpipe_0254 | $d = 0.0254$<br>$2 \le l \le 7$ | | 55,743 | 7 | $2.0953 \times 10^{-5}$ | $1.3675 \times 10^{-4}$ |
| SVC-1v | svc_1v | $-0.0005 \le q \le 0.0005$<br>$-7 \times 10^4 \le \Delta P \le 7 \times 10^4$<br>$0 \le v_1 \le 1$ | | 10,077 | 6 | $1.1351 \times 10^{-4}$ | $3.1619 \times 10^{-4}$ |
| SVC-2v | svc_2v | $-0.0003 \le q \le 0.0003$<br>$-5 \times 10^4 \le \Delta P \le 1 \times 10^5$<br>$0 \le v_1 \le 1$ | | 10,077 | 10 | $1.8831 \times 10^{-5}$ | $4.5144 \times 10^{-5}$ |
| SVC-2v | svc_2v | $-0.0003 \le q \le 0.0003$<br>$-5 \times 10^4 \le \Delta P \le 1 \times 10^5$<br>$0 \le v_1, v_2 \le 1$ | | 10,077 | 10 | $1.8831 \times 10^{-5}$ | $4.5144 \times 10^{-5}$ |
| PC | pc | $-0.001 \le q_{in}, q_{out} \le 0.002$<br>$10^5 \le P_{in} \le 2.5 \times 10^5, 10^5 \le P_{out} \le 4.5 \times 10^5$<br>$0 \le v_1 \le 1, 0 \le \omega_{pmp} \le 400$ | | 13,200 | 10 | $5.7994 \times 10^{-4}$ | $9.3825 \times 10^{-4}$ |

Firstly, the component model was built with the Flowmaster tool and connected to Matlab using a COM interface in order to create the training data by performing the computer experiments. For each surrogate model, the training was performed five times with the same training data, and then the NN model with the best training MSE (mean-squared error) was chosen as the final model.

Particularly for the VPIPE model, three different NN-surrogate models were created to form it, and it was done to be able to achieve better accuracy and efficiency in modeling. During the several trial-and-error iterations for creating a surrogate model of the VPIPE model, the inclusion of both pipe length and diameter into the input space turned out to be a less suitable decision since the response data from the computer experiment came with a very broad order of magnitude, which deteriorated the accuracy of the resulting surrogate model. In fact, the YP fluid system was made of pipes with only three different diameters so the approach of creating a separate surrogate model for each pipe diameter resulted in a good accuracy without sacrificing affordability of modeling.

Although most of the surrogate models had relatively large training data, the computation cost for both computer experiments and training the models was not considerably high, except for the CP model. For most models, the computation time for the computer experiment was well less than one hour and a single training took only a few minutes (based on a machine with an Intel Core II Duo processor and 2 Gbyte memory). On the other hand, the CP model, which had the most complicated configuration among the component models, needed significantly longer time -- about 4 to 5 hours -- of computer experiment than others mainly due to the computational cost of the Flowmaster tool.

### ***Simulation Settings***

The simulation ran with a discrete time step of 0.05 second, and throughout the whole simulation, the PC no.1 sub-net operated with a single pump turned on to a fixed speed of 200 rad/s (about 1910 RPM), while PC no.2 was in off-state. When PC no.1 was damaged, the controller on PC no.2 turned the pump in the PC no.2 on in order to continue the system-level operation. The 18 damage control valves were connected to the reference damage control model.

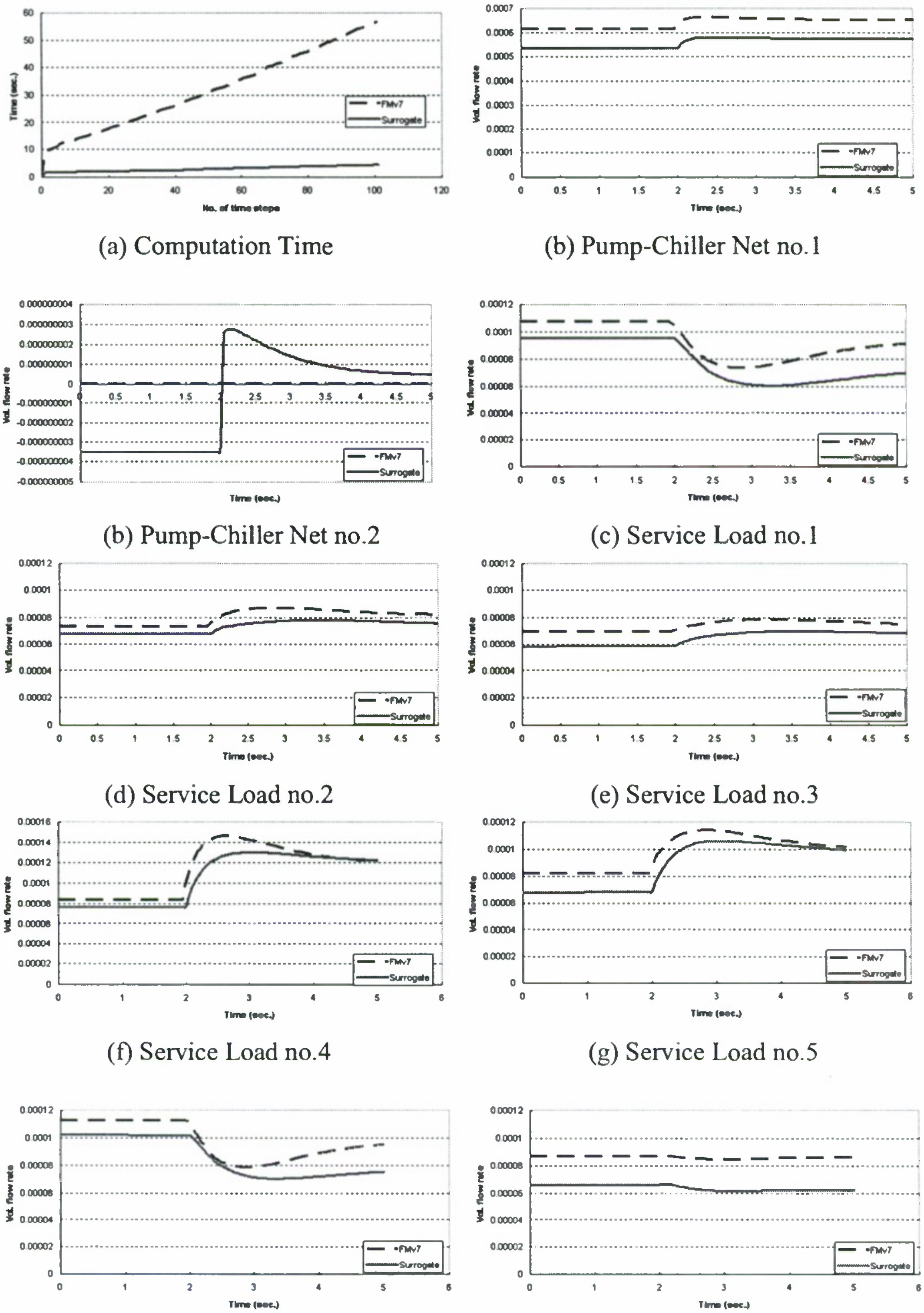
Two analyses were performed in this demonstration. The first was the validation of the graph-based surrogate model of the YP fluid system. In the first analysis, the simulations of both the Flowmaster and the graph-based models were executed with no damage control attached, and at the two second point of the simulation, the damage occurred at location (7,4,0) in Figure 30. Then the accuracy and the computational cost of the graph-based surrogate model were compared to the Flowmaster counterpart. As an additional setting to mimic the condition that the two models were run under an integrating framework, both the Flowmaster model and the graph-based surrogate model of the YP fluid system were linked to ModelCenter® using its script-wrapping support.

Then, a simulation-based damage experiment was performed with 28 different damage locations as the next step. The damage was set to be triggered at the one second point of the simulation, and the damage locations were set evenly distributed through the system and are given in Figure~\ref{fig:res2-table}. For each simulation run, the operational capability of the fluid system was quantified and plotted, as an example of how this model may help the system engineers exploring and building intuitions for designing a more resilient system.

For all simulation runs, the damage bubble representing the damage was set to the same radius of 0.7.

### ***Damage Model validation***

The analysis was performed with the simulation running for 5 seconds of simulation time, and the response comparison is shown in Figure 31. As in Figure 31(a), the graph-based surrogate model took 4.48 secs which was 12.6 times faster than the Flowmaster model, which took 56.48 secs. However, as a compromise to the great benefit in the computational cost, the graph-based surrogate model contained, errors and biases in its response based on the responses of the Flowmaster model, and this trait is in fact common to every surrogate model due to its nature, which is an approximation of a model. These errors and biases could come from the combination of various reasons, but the highest contributor might be the insufficient model accuracy of the PC surrogate model. Because of the high computational cost of generating the training data, the PC surrogate model was not created from a large enough training data set for its dimension and size of the input space, which led to more prediction errors of the PC surrogate model when in use.



(a) Computation Time

(b) Pump-Chiller Net no.1

(b) Pump-Chiller Net no.2

(c) Service Load no.1

(d) Service Load no.2

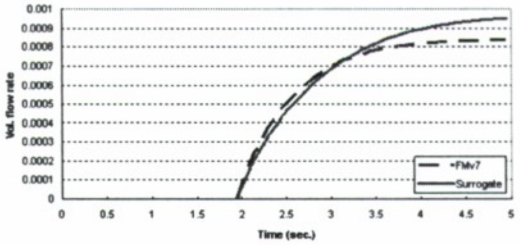
(e) Service Load no.3

(f) Service Load no.4

(g) Service Load no.5

(h) Service Load no.6

(i) Service Load no.7



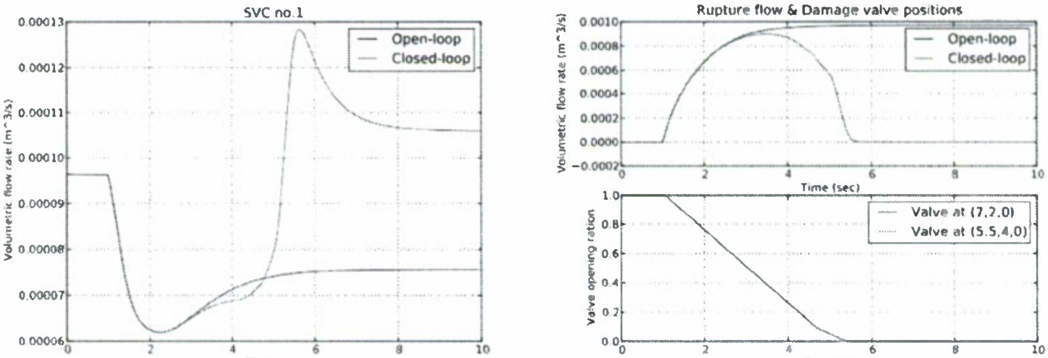
(j) Total Rupture Flow

**Figure 31: Comparison of the Responses of Flowmaster and Graph-Based Surrogate Models with the Rupture at (7, 4, 0)**

Despite of such a problem, the graph-based surrogate model would still be very useful, especially for the early phase of the design process. In practice, the Flowmaster model is not necessarily more accurate than the graph-based surrogate model when it comes to modeling in the early design stage because a large portion of the system detail is yet unknown or not decided at all. Instead, what is needed more in the early design stage is a computationally affordable model that lets a designer run a large number of analysis cases for exploring as large a design space as possible, meaning that the graph-based model can be a great choice to serve the purpose.

### *Damage analysis*

In the second analysis, the system was closed by the reference damage control, and each of the 28 simulations ran to 10 seconds of simulation time. Figure 32 shows the comparison of two of the open-loop and the closed-loop responses of the graph-based surrogate model with the same damage condition as the one in the first analysis. In Figure 32(a), right after the rupture occurred at the one second point of the simulation, the closed-loop system had a similar sudden drop of flow rate as the open-loop system, but eventually settled into a new recovered steady state. Although Figure 32 is only one of the 28 results with the different damage conditions, the behaviors should be similar to Figure 32.



(a) Flow Rate at SVC no.1 (b) Total Rupture Flow and Valve Inputs

**Figure 32: Comparison of Open-Loop and Closed-Loop Responses of YP-Fluid Model with Rupture at (7, 4, 0)**

In order to quantify and measure the system recovery performance for every simulation case, the operation capability rate (OCR) was defined in the following way:

$$OCR = \frac{w_1\tilde{q}_1^f + w_2\tilde{q}_2^f + w_3\tilde{q}_3^f + w_4 \cdot \max(\tilde{q}_4^f, \tilde{q}_5^f) + w_6\tilde{q}_6^f + w_7\tilde{q}_7^f}{w_1 q_1^o + w_2 q_2^o + w_3 q_3^o + w_4 \cdot \max(q_4^o, q_5^o) + w_6 q_6^o + w_7 q_7^o} \qquad (1)$$

and,

$$\tilde{q}_i^f = \begin{cases} q_i^f & \text{, if } q_i^f \le q_i^o \\ q_i^o & \text{, otherwise} \end{cases} \qquad (2)$$

where, $q_i^o$ and $q_i^f$ (i = 1,...,7) were the initial and the final values of the volumetric flow rate, respectively, at the 7 heat exchanger units, and $w_j$ (j = 1,...,6) were the weight coefficients for the six service loads in the system. In Equation (1), the terms $w_4 \cdot \max(q_4^o, q_5^o)$ and $w_4 \cdot \max(\tilde{q}_4^f, \tilde{q}_5^f)$ reflected the fact that the two heat exchangers SVC no.4 and SVC no.5 were a redundant pair serving a single service load.

The OCR is in a scale of 0 to 1 which represents a measure of how well the recovered system held its chilled-water delivery capacity from the level of the system before damage. Given the formulation of Equation (1), the OCR estimation strongly relies on the right choice of the weight coefficients which represent the service load priorities based on customer requirements, mission profile, and design philosophy, but in this analysis they were chosen by the author just as a demonstration purpose and given in Table 5.

A system's recovery performance should not be measured only using the final system state after recovery but also by how fast the system recovered. The OCR in Equation (1) was not formulated so just because the model was with the reference control model that has no control-induced delay in damage control efforts, but in the real application to the control development and test, the recovery speed must be taken into account in the formulation of OCR.

Figure 33 is the result of the damage analysis performed with the 28 damage cases.

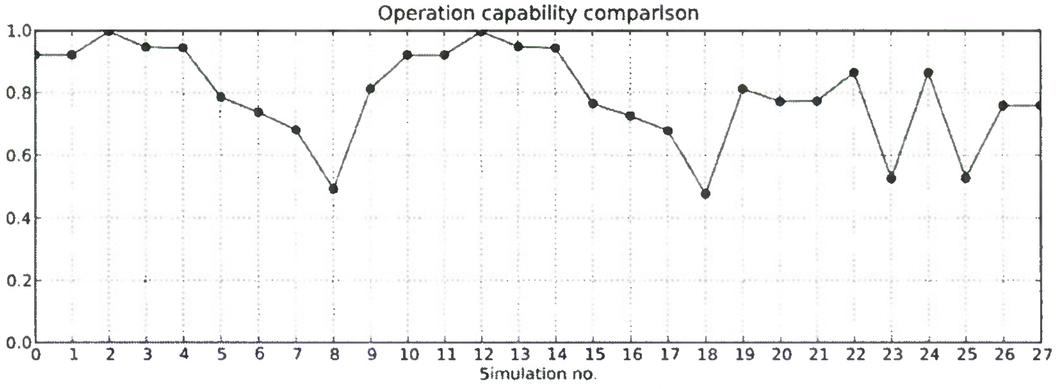


**Figure 33: Operation Capability Rates**

### Table 5: Component Failure Status

| | | Rupture location | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Starboard, lower | | | | | Port, lower | | | | | Starboard, upper | | | | | Port, upper | | | | | Aft | | Mid | | | | Bow | |
| Simulation no | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| Weight (1~10) | Components | (0, 0, 0) | (1.75, 0, 0) | (3.5, 0, 0) | (5.25, 0, 0) | (7, 0, 0) | (0, 4, 0) | (1.75, 4, 0) | (3.5, 4, 0) | (5.25, 4, 0) | (7, 4, 0) | (0, 0, 2) | (1.75, 0, 2) | (3.5, 0, 2) | (5.25, 0, 2) | (7, 0, 2) | (0, 4, 2) | (1.75, 4, 2) | (3.5, 4, 2) | (5.25, 4, 2) | (7, 4, 2) | (0, 2, 0) | (0, 2, 2) | (4, 1, 0) | (4, 3, 0) | (4, 1, 2) | (4, 3, 2) | (7, 2, 0) | (7, 2, 2) |
| | PC # 1 | ▒ | ▒ | ▒ | ▒ | ▒ | ▓ | ▓ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▓ | ▓ | ▒ | ▒ | ▒ | ▓ | ▓ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ |
| | PC # 2 | █ | █ | █ | █ | █ | ▒ | ▒ | █ | █ | █ | █ | █ | █ | █ | █ | ▒ | ▒ | █ | █ | █ | ▒ | ▒ | █ | █ | █ | █ | █ | █ |
| 2 | SVC # 1 | ▓ | ▓ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▓ | ▓ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▓ | ▓ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ |
| 6 | SVC # 2 | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | █ | ▒ | █ | ▒ | ▒ |
| 5 | SVC # 3 | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ | █ | █ | ▒ | ▒ |
| 8 | SVC # 4 | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | ▒ | █ | ▒ | ▒ | ▒ |
| | SVC # 5 | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | █ | ▒ | █ | ▒ | ▒ |
| 2 | SVC # 6 | ▒ | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ |
| 6 | SVC # 7 | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | █ | █ |
| | OCR | 0.92 | 0.92 | 1.00 | 0.95 | 0.94 | 0.79 | 0.74 | 0.68 | 0.49 | 0.81 | 0.92 | 0.92 | 1.00 | 0.95 | 0.94 | 0.77 | 0.73 | 0.68 | 0.46 | 0.81 | 0.77 | 0.77 | 0.86 | 0.53 | 0.86 | 0.53 | 0.76 | 0.76 |
| Computation Time | max | 6.384 sec | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | min | 5.326 sec | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | mean | 5.910 sec | | | | | | | | | | | | | | | | | | | | | | | | | | | |

▒ : Operating
▓ : Down
█ : Turned-off

In Figure 33, the average value of the OCR in the damage analysis was 0.80 meaning that the system's overall capability to recover from a single rupture was quite good, although the system still had room for improvement. The result has a few interesting patterns, one of which was that simulations 8, 18, 23, and 25 did not only yield the four lowest OCR values but also had all the same damage area, which was the mid-area of the system or more specifically, the bypass pipelines. This particular pattern in the analysis result clearly shows where to start for a more survivable and resilient system design. As a next step, a design engineer can create a number of design alternatives with different schemes of control strategy, valve placement, bypass pipeline placement, or service load locations, and then perform the same routine of modeling and damage analyses introduced here to the group of design alternatives in an automated manner.

### Future Research

Among various design problems for improving the survivability and resiliency, the next analysis will describe an example on how the graph-based surrogate model can be used for finding a better damage control valve placement and the effect of the different amount of valves to the recovery performance, especially on the middle area of the system. The analysis will be based on an extensive number of damage simulation with the full factorial variations of the valve placement, with different total amounts of the damage control valves at the mid to bow area of the fluid system.

## Task 4: Human in the Loop Control

### *Introduction*

With the integrated modeling and simulation environment, the behavior of the system can be studied. To better understand the simulation results, visualization capability is required. The Human Machine Interface (HMI) started as a simple visualization tool for observers to identify and understand emergent behaviors of the complex system, such as IRIS type systems, as the system consists of multiple subsystems. The tool allows for rudimentary user interaction to see how behaviors might be influenced by human interaction.  Since the HMIs conception, the tool has spurred many other ideas and questions. These questions challenged how engineering tools are designed and used.

They challenged the nature of useful engineering knowledge for design purposes. In the process of investigating solutions to these challenges, the HMI is transforming into a design tool seeking to increase the accessibility of engineering knowledge by pooling the strengths of distributive systems. This is referring to the distributive nature of the design process of large complex systems.

By default large complex systems must be decomposed in hopes that the system may be understood by its parts. Engineers attempt to maintain both the macro and the micro perspectives of the system. Nonetheless, experience teaches us that these attempts are generally expensive and risky. Risk is introduced as decisions are made. Knowledge may mitigate risk, thus the more that is known when a decision is made the less risk the decision might incur. The catch is that knowledge is built upon decision. One cannot understand the system as a whole until it can be understood by it causes, which are determined by the interacting parts.

The HMI might be able to contribute to a solution. From the beginning the HMI was design as a web-based tool providing a level of abstraction between the user and the services. The service in this case is a remote simulation environment. This abstraction would enable engineers outside the IRIS project to study a complex system in a hands-on manner, interacting with the simulation, and observing behaviors. By introducing key levels of abstractions between users and services, and services and other services, the complexities of knowledge building and designing might be controlled. Thus reducing risk during the design process.

Fundamentally IRIS is an exploration: it is an activity to build engineering knowledge for a specific class of systems. In its conceptions every decision requires a rational process to substantiate it. However, the design needed a starting point, a beginning to form a foundation for knowledge. This prerequisite transformed the project into an infrastructure project and an experimental plan designed to act as a first step to understanding. What was hopped to be learned was a true appreciation of the underline causes behind the behaviors of the integration of intelligent systems.

## *Objectives*

The vision behind the HMI could not be realized without some guiding objectives. Each objective has some roots from observations of projects from a variety of disciplines. The notion is that there existed a generalized solution to each phenomenon that project hoped to address. The objectives broke down into four categories:

1. Real-Time Operation
2. Data Fusion
3. Visual Analytics
4. Dynamic Data Sets

Many aspects of the objectives are being tackled in small steps. The following sections will reveal more as to why these objectives have been chosen and how they are being addressed.

## Real Time

In order to properly present a simulation intended for human interaction, the simulation environment and the HMI must be able to maintain a real time performance requirement. This helps human operators obtain a feel for the responsiveness of the system to external stimulus. This burden is heaviest on the simulation environment, but it does mandate that the HMI must have the feel of a locally executed application on the desktop.

The HMI client takes advantage of asynchronous communications and data pre-fetching to achieve this goal. The theory being that it will be network lags that are the most encumbering. In general this belief has held true. However it has been noticed that older computers do show some lags do to the load of a full screen flash application. Often older computers are coupled with new monitors with higher resolutions, or for some other reasons they do not meet the minimum video performance required. This will lead to a hardware requirement specification that will be released with each version of the client.

## Data Fusion

The development of the HMI has created an interesting opportunity in the realm of design science. On the one hand the HMI has a very tight integration with simulation environments, specifically those utilizing Model Center. On the other is an application framework for analyzing data. In between is a database. Collaboration systems for design purposes have been a long standing interest at ASDL

- Distributive Design
- Distributive Modeling
- Distributive Simulations
- Etc.

Data fusion is a step beyond data integration. Data integration is the process of merging two or more data sets into one, while data fusion suggests that more can be learned from merged data sets after a reduction process. People perform data fusion when they abstract meaning from multiple data sets and then determine meaning or consequence from the combined abstracts. A system with a service orientated architecture can be utilized to perform data fusion tasks in a distributed fashion. Only this fusion process does not need to be limited to raw data. It could be applied to designs, models, and simulation environments. The key is to maximize the potential benefits of any effort by keeping modularity and reusability in mind. Object orientated programming achieved this at the application level. The next step is service orientated, network centric architectures of both data and software.

Historically applications and data have been treated as static entities. Unfortunately the reality is that these entities change frequently. The applications change. The models change. Software in general will change. Change in fact has become the problem. The problem was created after computer systems became decentralized during the beginning of the era of the personal computer. Decades later with the advent of the internet computer systems are beginning to resemble a hybrid between centralized and distributed systems. This hybrid if realized can bring the information ages into a useful collaboration environment.

The vision of this objective is to explore the nature of the hybrid centralized and distributed system model and the implementations it may have on the design paradigm for complex physical systems.

### Visual analytics

Visual analytics has been described as the science of analytical reasoning facilitated by interactive visual interfaces. Recently attention in this area has been on the rise due to a strong interest in the subject from the department of homeland security. Visual analytics has the potential of enabling the processing of an overwhelming amount of disparate and conflicting data.

The design process is an act of analytical reasoning and the dimensionality of complex physical designs is overwhelming without the proper tools and techniques. The HMI is a visual analytics tool in that it was design to facilitate the design process using an interactive visual interface. The server further supports this role by supporting tools such as JMP from SAS and Microsoft's Excel.

### Dynamic Data Sets

This aspect refers to the use of tools designed to aid decision making, i.e. the so call calculator tools or sometimes call dashboard tools that are often utilized to help decision maker with making wise strategic decisions. These tools attempt to present information in such a way that it might be meaningful to a decision maker. This information is often based on some collected data that if not refreshed becomes very static. The HMI as an objective hopes to produce an environment for decision makers with real-time data.

# *Progress*

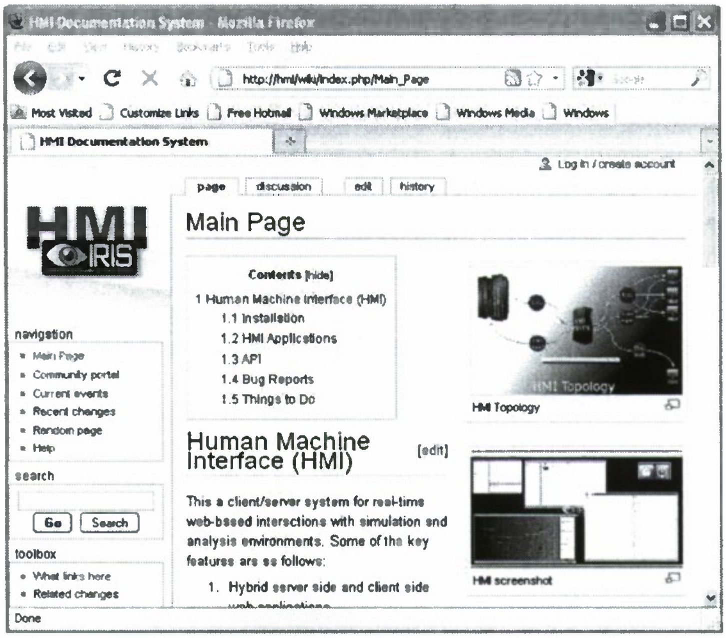
## Documentation System

#### ***Based on MediaWiki Project***

The MediaWiki Project has gained a strong presents in the web community. Not just within open source circles but the general public at large. This is primarily due to the success of Wikipedia which is based on the MediaWiki project. The underline concepts are based on the basic world-wide-web with one exception. MediaWiki opens the doors for contributors to provide content to the system.

#### ***User-based and Developer-based Contributions***

The HMI server has been equipped with a wiki system similar to that found at Wikipedia, as shown in Figure 34. Only here the wiki's purpose is to document the HMI. It has always been the intention for this project to enable users to further develop the capabilities of the HMI. So it only makes sense that users should also be able to contribute to the documentation.



Main Page

Contents [hide]

1 Human Machine Interface (HMI)
1.1 Installation
1.2 HMI Applications
1.3 API
1.4 Bug Reports
1.5 Things to Do

Human Machine Interface (HMI) [edit]

This a client/server system for real-time web-based interactions with simulation and analysis environments. Some of the key features are as follows:

1. Hybrid server side and client side

Figure 34: HMI Wiki

## Virtual Machine Based Builds

### Based on Virtual Box (Open Source)

Virtual Box is an open source x86 virtualization package. In essence one could create a machine (computer) virtually, configure it and redistribute the machine with few strings attached. This creates an ideal platform for engineers to build within, without having to have too much regard for the environment the virtual machine will be running in. Server virtualization is become quickly adopted in general for many advantages they afford, including but not limited to the ease of distribution, backup, and management.

### Easy Distribution

Once a virtual machine is configured and running it is a simple process to export the machine and burn to a disc or USB drive for easy distribution. The virtual machine encapsulates all the software required for the servers operations. The installation process is again another two step process. Download and install the virtual box software, and them import the virtual machine. Once the machine has been imported, press the start button and the system is up and running.

### Easy Snapshot Backups

Since the actual virtual machine is little more than a few files the entire machine, backups are simple. Snapshots are an easy way to protect the system from changes. At anytime (even while the system is running) a snapshot can be taken of the system. At which the system could be rolled back to any given snapshot at anytime. To protect against a hardware failure simply shutdown the virtual machine and export it to a backup location.

### *Low Resource Requirements*

Currently the virtual machine is running Ubuntu server 9.10 with a standard LAMP (Linux, Apache, XML, and PHP) installation. The HMI is then installed on top of the LAMP configuration. The virtual machine is configured for 512 megabytes of RAM and 30 gigabytes of disc space. This configuration was design for some growing room. Currently the actual disc space being used is less that 900 megabytes and the server will run with less than 128 megabytes of RAM.

## Model Center Plug-in

### *Two-way communication*

The new Model Center plug-in was only one way until now. Since the first version of the HMI the structure of the information sent between the simulation environment and server has changed significantly thus requiring new parsers and encoders to be written. The new implementation supports the XML dialect called dashML. The plug-in is written in java script using a flexible extension of the object class to handle all the encoding, decoding, and communications with the HMI server.

### *Built-in logger*

The new plug-in now has a new logging capability. Each event is logged and reported back to model center for verification purposes. This feature allows a user watching the simulation environment to quickly troubleshoot issues with either integration or networking. In addition to the logs the plug-in also reports the actual data being sent to the HMI server and the actual data received before parsing.

### *Built-in error handler*

The plug-in also has an error handler which should prevent the plug-in from ever crashing the simulation environment. Once an error is detected it is logged with an explanation, and the plug-in attempts to exit gracefully.

### *Stream-line work flow*

The work flow is condensed to simply specifying which variables are to be sent and which are expected back from the server. No more details are required from the user, making communication with a remote service as easy (if not easier) as working with a completely local environment.

### *Fully Documented*

The plug-in API is completely documented on the HMI wiki system. The documentation includes examples and explanations for each method and attribute currently in use. As mentioned earlier the wiki allows for the user to augment the documentation as appropriate to clarify for future use or customizations.

### *Future Work*

The items in this section have been reported in the past quarterly report in detail. The details will not be repeated here, however a short list is being carried over as a reminder of task being purposed for future work. The list is as follows:

- More advanced error handling
- Categorize errors
- Hybrid logging system (between the client and the server)
- Scalable vector graphics renderer
- Server side SVG compiler

It is notable that these points are having an influence on work that has taken place this past quarter. The ModelCenter plug-in which has been described above includes both the error handling and the logging capabilities.

# Task 5: A Methodology for Improving System Effectiveness in Resilient Systems Design

## 1. Introduction

Traditional design approaches are based on optimizing naval systems for performance, based on a very limited number of mission scenarios. While the traditional design approach is conceptually fairly simple and straightforward, it does not really address any issues regarding the uncertainty around naval system mission requirements, environmental condition or even the capability of the system to perform as designed under real operations. A robust solution will represent a system that in theory would be better prepared to perform multiple mission acts and withstand a larger spectrum of unexpected events. At the same time, prescribed design performance might not be optimal, in order to compensate for the multi-mission capability (e.g., preferred extra weight for redundant systems over maneuverability).

One of the main objectives of IRIS is to deliver a conceptual design methodology for more survivable and mission effective ships. However, the question that remains at this point is how exactly the multi-mission capability and the enhanced survivability are enabled. Typical survivability enhancement features, such as component redundancy, separation and shielding are immediate techniques that can be properly applied to the design based on conceptual sizing. Yet, there is no standard design method that determines the extent of such enhancements and the type of survivability that each enhancement would seek to improve (susceptibility, vulnerability or recoverability). The current United States Navy standard is primarily determined by the Survivability Design Handbook for Surface Ships (OPNAV P-86-4-99) and according to this procedure, survivability is improved by focusing only on vulnerability, implying that susceptibility reduction and recoverability improvement are expected consequences of the former.

In order to extend the state of the art to address all types of survivability, there is a modern philosophy recently emerging and seeks to address the aforementioned concerns.

Resilience engineering is a philosophical framework that encompasses a collection of concepts around safety management and engineering. Some of them focus on understanding threats, accident and damage propagation, as well as how a system should be designed to conform to changes that occur around it, for the purpose of withstanding adverse effects and maintaining its mission effectiveness.

The fundamental research question regarding this initiative would be *how to improve the design the system, so that system effectiveness through survivability is maximized for a given set of scenarios, which will include system damage and/or restoration events.* Moreover, it can extend to consider how the philosophy of resilience engineering can translate into a systems engineering method, involving various aspects, such as accident and damage modeling or system functionality and possible enablers, in order to fit into the bigger picture of more survivable systems in a highly uncertain mission environment.

## 2. Deliverables and Discussion

Three main research areas have been identified as supporting work to this task. The first thrust involves the clarification of key words and concepts and the theoretical framework, which the methodology will be built upon. The second thrust includes all the efforts for obtaining a suitable modeling and simulation environment, given the revolutionary nature of the underlying concepts and premises. The third is the development and the demonstration of the methodology, using a baseline naval architecture.

### ***Research objectives for improving safety***

The main objective of the first task is to further investigate the concept of resilience in the context of system safety and suggest a complete framework for assessing resilience in systems engineering. Up to present, there have been some attempts on quantification of resilience and most of them are heavily based on the particular discipline of practice. A resilience assessment framework would be instrumental and highly supportive for any effort on integrating system safety as a product and process characteristic in early conceptual system design. Moreover, it can extend to consider how the philosophy of resilience engineering can translate into a systems engineering method, involving various aspects, such as accident and damage modeling or system functionality and possible enablers, in order to fit into the bigger picture of more survivable systems in a highly uncertain mission environment.

Before the proliferation of the concept of system resilience and the incorporation of the philosophy into a design method, a case needs to be made for demonstrating the merits of resilient systems design over the current state of the art. With Robust Design and Multi-Disciplinary Optimization (MDO) being the most renown approaches in design for affordability and performance, and reliability/survivability-based design being the most prominent methods for designing for system safety, there should be a clear indication how a prospective methodology for more resilient systems would be an improvement over robust and survivable designs. Thus, from a big picture perspective, a resilience assessment framework would also greatly serve in investigating the benefits and detriments of robust and resilient systems, especially with respect to safety, survivability, mission effectiveness and ultimately obtain the cost-effectiveness tradeoffs.

Summarizing the aforementioned research objectives, the fundamental research questions that need to be addressed are:

- If a resilient system did exist, how would it look like and behave in a dynamic environment?
- How a resilient system does differ from a SoA survivable or safe system?
- What framework for assessing system resilience can be selected and developed in a way to test system robustness and the resilience characteristics of the system?

The fundamental hypothesis that has been formulated for testing and as an attempt to answer the above questions, is that *an existing analytical framework for survivability assessment can be extended to account for dynamic system responses, including the effects from emerging behaviors and changing operating conditions and requirements.* It would then adhere to the premises of resilience engineering and support the evaluation of system resilience. Part of this effort would be to characterize the behavior of multidisciplinary and interdependent system architectures for different classes of disturbance events. Given the observed patterns of system behavior, a set of resilience metrics are proposed, essentially capturing the three types of system survivability (susceptibility, vulnerability and recoverability) with respect to the system's dynamic responses. System functionality determines the dynamic responses of highest interest, such as mission performance outputs (e.g. mobility, stability) power delivery (e.g. electric or mechanical power) and system health monitoring (e.g. cooling and temperature levels, structural integrity etc.). Effects from emerging and unexpected adverse behaviors due to changing operating conditions and requirements should also be captured by the framework and included for the resilience assessment. For the purpose of demonstrating the effectiveness of the proposed framework, a simple problem has been devised, that incorporates dynamic behavior fairly similar to larger-scale heterogeneous complex system.

## ***System safety and risk***

System safety is known as the *"sum of all accidents that do not occur"* [4] or that a system is safe if there are not any accidents affecting its normal operating conditions. This presumes that the only risks around the system's operations would always result into a certain set of emerging accidents. In another attempt to define safety, Hollnagel has described safety as *"the state of the system at which nothing unwanted happens"* [4]. There may be several cases where there is something "unwanted" that actually happens, yet the system remains in a safe operating state and system safety is not threatened overall. On the other hand, an unsafe system state can emerge without the occurrence of anything that might be unwanted. This more general definition also supports a common perspective on safety in different parts of the organization e.g. aircraft maintenance and the occupational safety and health office [10] .

A reworked definition links safety to *"freedom from accidents or losses",* extending the lack of safety from accidents to losses as well, based on the premise that absolute safety does not exist, thus it should only be defined in terms of acceptable loss [17]. This would offer a basis for a quantitative framework of evaluating safety, namely through the quantification of the loss from a nominal safety target, whether that really refers to a system's performance measure, utility function or behavioral response. If safety can be visualized as a continuum [17], then a continuous function can be defined that represents the system's loss.

Safety is dynamic, not being a property of static parts but the outcome of complex processes. Accidents occur when external disturbances and dysfunctional interactions between system components create a situation that gets out of control. With this perspective, safety can be viewed as a control problem [16]. The main function of safety management is then to control system and sub-system process performance, under the effects of operational risk and mission uncertainty.

But how exactly various risks are formulated? Given the uncertainty around system operations, maintenance and reliability, the risk of experiencing an accident is linked to safety as a means to quantify operational uncertainty. Risk is typically measured as the probability of a certain set of events occurring, originating from faults and subsystem or system failures. Moreover, several threats, hazards and risks, either external or internal can emerge due to challenging mission expectations or changes in the system's environment [11], [4]. The appearance of new hazards is a common risk factor, especially as more revolutionary technologies and solutions are introduced for mass consumption with a continuously decreasing average time of conversion of technical discovery into commercial product [17]. Popularity of modern technologies can also become a risk factor. One can understand that the increased availability and popularity of modern technologies has increased the exposure to hazards. For instance, modern automobiles are safer by design and include more safety features, yet popularity of cars has also increased dramatically, thus increasing the chance of a driver being involved in a motor vehicle accident. Chemicals and drugs can protect and provide relief from many types of illness, virus or disease. However, increased usage or misuse can lead to other risks due to drug side effects or the creation of resistant microbes [17].

Increasing complexity in modern engineering systems can be considered as a consequence of extensive usage of modern technology solutions. Which in turn, can be the consequence of increasing societal demands on system capability or the application of government enforced regulations or policies. The latter trend is also known as the increase of scale and centralization and has lead to the development of large scale complex systems, e.g. spacecraft or naval combatants. High technology systems are often made up of networks of closely related subsystems. Conditions leading to hazards emerge in the interfaces between subsystems, and disturbances progress from one component to another [17].

Even if safety does not seem to span the entire threat spectrum as an inclusive discipline, safety is still considered as the starting concept, off which other concepts have evolved, and ultimately branched out. This process is very common in science, as the evolution of needs follows the change that continuously occurs in modern societies and nature. A similar development is observed for survivability, with researchers lately tending to focus on one aspect of survivability (susceptibility, vulnerability or recoverability) only, rather than collectively attempting to assess survivability from a high level perspective. On the other hand, for all the concepts relevant to safety engineering, a common basis can be identified that will be instrumental for the definition of evaluation and assessment frameworks and can be summarized in the following set of three dimensions:

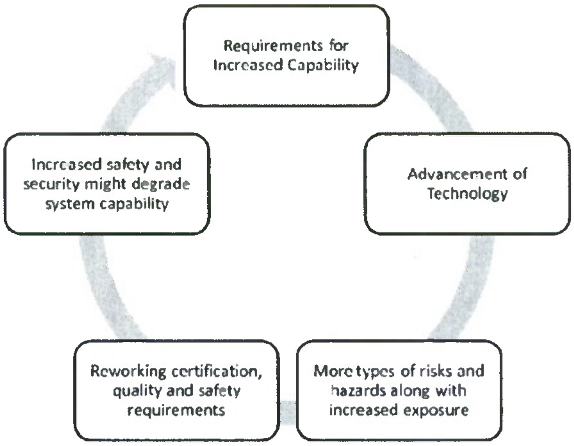

Requirements for Increased Capability
Advancement of Technology
More types of risks and hazards along with increased exposure
Reworking certification, quality and safety requirements
Increased safety and security might degrade system capability

**Figure 35: From system requirements to risk and the need for improved safety**

***Safety and other similar concepts***

As evidenced by its broad definition, the concept of system safety is not unique within its own context, but can also be effectively discussed and considered by other similar concepts, such as system reliability, security, survivability and resilience. Even with a more thorough approach, overlap with safety can be discovered in the less affiliated concepts of availability, maintainability and fault tolerance. There is absolutely no reason to only base a safety assessment solely on one concept and dismiss the others, given that they all exist for capturing different parts of the safety management spectrum. However, it would be useful to briefly investigate the underpinnings of each concept, for drawing the lines that distinguish one from another, but also be aware of the amount of the overlap among them.

Based on their commonly acceptable definitions, Richards has offered a clarification of the boundaries and the overlap of safety, security, survivability and reliability [12]. For this purpose, the type and the origin of the possible risks and hazards has been considered in order to construct a 2D visual schematic of what combinations are expected to be covered by each safety related discipline. For the threat type, there can be typically two types, *natural (non-intelligent)* and *malevolent (intelligent)* threats or resulting hazards. Regarding the origin, there can be either endogenous or exogenous threats. If safety can be considered as "freedom from accidents or losses" [17] and given that accidents in this context are usually attributed to natural threats (unexpected and non-intelligent), it is concluded that safety can refer to either endogenous or exogenous threats.

**Security**

Security is a relevant discipline that appears to be slightly more focused to malevolent threats than safety. According to Leveson's definition, security is "a system property that implies protection of the informational, operational, and physical elements from malicious intent" [14]. Security can also refer to different applications and entities. Except for its applicability to standalone entities, such as physical devices and computers, security applies also in networked systems, privacy and information, as well as national infrastructures or operations. Security can be assessed for either endogenous or exogenous threats, yet it is restricted to actions that all seek to affects a system's normal operating conditions, goals, integrity and durability. Moreover, security typically deals with protecting against losses, system performance degradation and furthermore to apply regulations and requirements imposed by governments or relevant entities for enhancing protection. In contrast to safety, security as mentioned earlier is focusing on malicious, intelligent and intended actions against systems or entities, regardless of whether they are endogenous or exogenous, whereas safety includes non-intelligent, well-intended or natural actions as well.

### Reliability

Reliability is the probability that a component will perform its intended function "for a prescribed time and under stipulated environmental conditions" [17]. As opposed to safety and security, reliability is primarily focusing on how system operations are affected by internal component failures. In other words, reliability engineering only involves endogenous failures and risk that might lead to accidents, mainly due to natural, non-intelligent faults that may also include the effects of maintainability, aging and deterioration. In the context of the last three modes, failures can be exogenous to some extent, given that maintenance or aging are significantly controlled by change occurring externally, thus reliability engineering is considering exogenous factors in that fashion.

### Survivability

Survivability is a concept that to a great extent it attempts to collectively bring the concepts of reliability, safety and security together. Historically, the concept of survivability was initiated and originally applied by military systems engineers [4] with definitions and related background mainly sourced from the aerospace engineering domain, yet survivability has proliferated in several other scientific and engineering domains as well. Indeed, the concept of survivability is quite broad and has been applied in different engineering domains and no unique and global definition exists. However, as it has been stated earlier, there are several commonalities concerning the concept application and any attempt towards a more unified definition would definitely make sense from a technical perspective. Survivability can also be considered as a design attribute in naval or merchant ship design, in automotive systems and architectures, in software engineering and IT support systems and networks, or in civil engineering applications. Science based disciplines, such as biology, computer science, systems network theory, etc., also make use of the concept of survivability.

There is no doubt that a wealth of literature exists concerning the science of assessing survivability and the pursuit of more effective strategies and technologies that seek to improve a system's resilience when it is challenged to withstand a threat. However, it has been observed that traditional design methods do not include survivability as a design objective in the early conceptual design phase. On the contrary, existing State of the Art survivability-based methods mainly rely on suggesting survivability enhancement strategies [4].They can be effective, yet applicable in the late design stages as means of improving the existing design after prior unsatisfactory survivability performance. Other methods are merely based on directives and standards defined by domain established organizations and authorities. While the do a great job in formulating an "ideally" survivable system indicating the direction of improvement, they rarely take into account the system mission objectives and the threat environment. Additionally, tradeoffs are not thoroughly examined and for those that do that, it usually involves cost-effectiveness considerations.

*Susceptibility* [4] is the "probability that the system experiences a direct hit or secondary hit effects through an attack by its environment". It refers to the inability of a system to avoid experiencing disturbances for at least one of its basic functions by one or more threat effects in the pursuit of its mission [4]. While this definition has been based on literature regarding military aerospace vehicles, all systems can be susceptible to a threat. Crash avoidance for instance, is the equivalent of susceptibility for automotive systems, while similar definitions hold for naval or IT systems, etc. Susceptibility is an important component of survivability, in the sense that can sometimes a threat can be entirely avoided and furthermore ensuring that system vulnerability should not be challenged at all. Indeed, survivability equation reveals that vulnerability is a conditional probability that depends on the outcome of a threat (implying that threat has affected the system's mission), while susceptibility is a probability that solely depends on whether the threat was encountered or avoided.

Ball claims [4] that the level of system susceptibility in a threat encounter is dependent upon three major factors: the threat, the system, and the mission scenario. A threat can be described by its characteristics, its operations, and the effective impact on the system. The system can be exposed to the threat by its observables (or detectable signatures). More exposure can follow by any countermeasures used to defend from the threat. Countermeasure effectiveness (or system effectiveness at a large extent) is determined by system performance capabilities and self-protection armament. The mission scenario entails the physical environment in which encounters occur, the threat deployment and activity, and the system mission description and tactics.

*Vulnerability* is defined [4] as the "conditional probability that the system is killed after it experiences a direct hit or warhead fusing through an attack by its environment". It refers to the inability of the aircraft to withstand the damage caused by the man-made hostile environment, to its vincibility and to its liability to serious damage or destruction when hit by enemy fire [4]. The more vulnerable a system is, the more likely it will be killed when attacked by one or more damage mechanisms. Needless to explain, for similar reasons that all systems are susceptible to threat, the same systems are vulnerable to threats, given that they failed originally to avoid the threat. At the scenario phase where a system is experiencing the impact of a threat, system overall vulnerability depends upon the subsystem vulnerability. However, this does not necessarily mean that a system comprised by low vulnerability components, will demonstrate low vulnerability to threat altogether. The truth of this statement can be further enforced, especially if the system is complex and exhibits emergent behaviors under different environmental conditions or threats.

In an attempt to provide a general and inclusive definition, survivability could be considered as the "ability of a system to preserve itself and its mission under the occurrence of disturbances". In an attempt for a more technical and general definition, Richards et al. have defined it as the ability of a system to minimize the impact of a finite disturbance on value delivery, achieved through either [12] the satisfaction of a minimally acceptable level of value delivery during and after a finite disturbance, or the reduction of the likelihood or magnitude of a disturbance.

In contrast to safety and security, survivability is transferring the emphasis on the system's behavior and recovery mechanisms. Susceptibility as part of survivability covers the part of evaluating how well a system can avoid or defer a threat, yet most recent survivability studies and assessment methods focus on vulnerability primarily and recoverability as a secondary concern. Given that the majority of threats usually translate in exogenous actions and changes around the system, survivability assessment methods often dismiss endogenous factors, unless it has to do with systems that internal changes are more critical than any external. A differentiating feature against safety is that the latter mainly deals with non-intelligent and natural threats, whereas survivability usually assumes intended, intelligent and malevolent actions. In that sense, survivability is more closely related to security in that both are concerned with malevolent environments, where intended and intelligent action takes place. The distinguishing factor on the other hand is that security includes threats of any origin, while survivability excludes endogenous threats, while including all types of threats. All of the above observations can be visually summarized in Figure 36 with the overlapping relationship of reliability, safety, survivability, and security, represented as a Venn diagram representation [12].

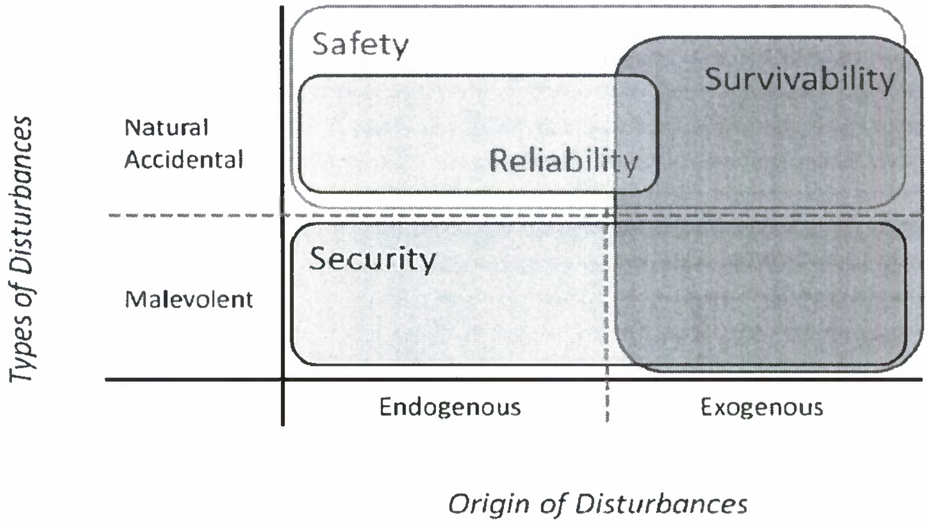


**Figure 36: Boundaries of Safety, Security, Survivability and Reliability [12]**

### ***Traditional approaches and SoA***

Returning back to the problem of safety itself, there is no best approach for improving and enhancing safety, given the volatility of the uncertainty due to new technologies and safety requirements. However, the necessary safety levels can be determined by understanding, estimating and managing risk in order to either eliminate the occurrence of accidents, or to mitigate the consequences if an accident is unavoidable. The most effective system safety processes seek for a more proactive and aim in preventing, eliminating and controlling hazards and risks through a collaboration of key engineering disciplines and product teams. Hazard analysis is always necessary for identifying risks and for specifying design safety features and procedures to strategically mitigate risk to acceptable levels before the system is certified.

*Safety Management* is the discipline that makes use of available approaches and techniques for improving system safety. For most of these techniques, the objective is not only about designing systems that can survive accidents, but also preventing them from happening. Typical mission of a safety-engineer is to perform a fault analysis study, propose safety requirements in design specifications and eventually demonstrate that a given design is safe. Some examples on safety management techniques are threat analysis, common cause failure (CCF) analysis, fault tree analysis (FTA), failure mode and effect analysis (FMEA), risk identification and estimation, design for reliability and survivability, etc.

Today's State-of-the-Art is strongly relying in Risk-Benefit analysis Risk identification and Rick assessment methods [17], yet they are typically being applied after the detailed design phase. Given that risk can be quantified, the acceptable level for risk must be selected, yet accounting for any other possible sources of risk that are linked to unavailable or non-existent knowledge. Baseline development and prototyping are necessary because it's impossible to measure risk before system is built and this is the major reason why this task is performed during the last design phases. Accident investigation methods usually include Fault/event tree or common cause (CCF) analysis and Accident and damage modeling through physics-based modeling and simulation. The State-of-the-Practice is to design system architectures with safety in mind (not quite by having safety fully integrated in the design process) and make sure that it is sized for all the necessary technologies that are relevant to safety, in order to satisfy safety regulations and instructions. In many cases, technologies need to be applied in the post-detailed design which implies that certain modifications might need to incur on the architecture for becoming compatible with retrofitted technologies.

Regarding the design methods that integrate the above techniques, each method is focusing on a certain aspect of safety, accordingly to the system's design or certification requirements. In reliability engineering (related to structural design or manufacturing) certain methods for Reliability-based Design Optimization (RBDO) have been applied, yet these are focusing on the long term health of the system, by making it robust to unexpected, endogenous, unintended and non malevolent failures. Design methods for security include systematic policymaking, cryptography-based approaches for reducing the risk of external intrusion or breech, physical shielding or multi-stage system component protection. Survivability-based design has proliferated in parallel, yet focusing on a subset of threats that is almost irrelevant to that of reliability-based design. The most common approaches on this realm is to include the effects of existing state-of-art techniques and technologies for improving survivability (e.g. redundancy, automation, separation and zonal design) and size the architecture including a subset of them subject to cost and to their expected prescribed effects in hypothetical threat scenarios.

So what is wrong with the SoA? Traditional design approaches would focus on performance, while modern approaches will seek for more robust solutions, either through enhancing safety or adding automation or intelligence. Beyond the development of methods that allow for the discovery of such solutions, studies have also be proposed to investigate the tradeoff of cost vs. effectiveness across solutions representing different underlying design philosophies. Safety requirements for system certification are typically satisfied in the latter design stages with the proposition of technologies and additional equipment that is being retrofitted in the architecture. Systemic view of accidents can deem the typical PRA methods as inadequate to represent complex concurrences of multiple factors that lead to accidents. Graphic representations in PRA/PSA methods cannot capture the dynamic behavior of a complex system. Complex systems are dynamic and their dynamic stability may change into dynamic instability, thus systems must be dynamically stable.

### ***Design for Resilience: The need for a theoretical framework***

Resilience engineering can offer insight and research directions that may lead to answers regarding the design of more safe and survivable complex systems. According to the systemic view of how accidents occur, one can infer that a resilient response by the system would include the ability to efficiently adjust to non-favorable influences rather than to resist them. Such ability could be embedded as collection of internal functionalities and be the basis for certain active features for susceptibility/vulnerability reduction and recoverability increase. Automation and networks of sensing grids and information distribution might be possible enablers for enhanced reconfigurability and would lead to the essential functionality of a resilient system. In other words, a resilient system is expected to adjust its functioning prior to or following changes and disturbances so that it can go on working even after a major mishap or in the presence of continuous stress.

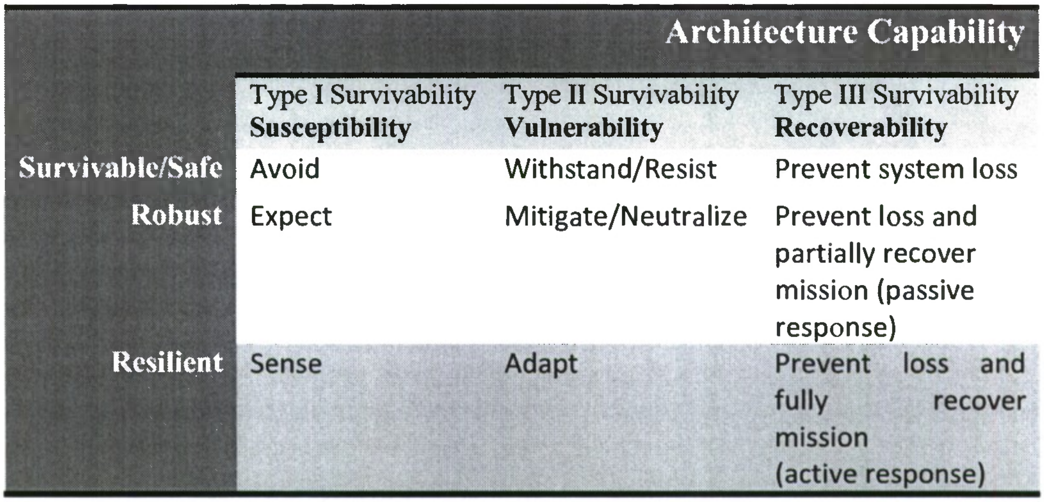
However, the traditional definition of resilience, as given by Holling (1973), contains a limited perspective on how a system can be safer [8]:

*"Resilience is a measure of the persistence of systems and their ability to absorb change and disturbance and still maintain the same relationship between populations or state variables"*

It is implied that a system would be less vulnerable and more recoverable by being capable of absorbing adverse changes that affect its normal operating conditions. In other words, resilience at this point is presented as a measure of robustness. However, the vision of a resilient system hints to a set of expected system features and characteristics that go beyond the characteristics of a robust system. Thus, a resilient system is a robust system, yet a robust system is not necessarily resilient.

As a first attempt to distinguish between robust and resilient systems, Table 6 has been constructed. For three conceptually different designs of the same system (baseline safe, robust and resilient) a breakdown of expected system functionality is provided, according to existing definitions for all three types. While safe systems mainly aim towards preventing system (including human life) without any provision on the mission by avoiding or resisting threats, robust designs additionally seek ways to partially recover the mission after assuring system survival. Instead of employing actions to withstand the adverse effects of a threat, robust systems should just be inherently insensitive to change by design, thus without requiring any particular attention for avoiding/resisting the threat.

**Table 6: System functionality for all three types of survivability**



| | Architecture Capability | | |
|---|---|---|---|
| | Type I Survivability<br>**Susceptibility** | Type II Survivability<br>**Vulnerability** | Type III Survivability<br>**Recoverability** |
| **Survivable/Safe** | Avoid | Withstand/Resist | Prevent system loss |
| **Robust** | Expect | Mitigate/Neutralize | Prevent loss and partially recover mission (passive response) |
| **Resilient** | Sense | Adapt | Prevent loss and fully recover mission (active response) |

For the resilient system, it is additionally expected that it is possible for threats and system status to be sensed, with the ultimate purpose to actively recover the mission after system loss is prevented. Adaptability to change is key distinctive feature of the resilient system compared to safe and robust designs.

Hence, in order to expand on the boundary allocation of disciplines as presented in Figure 36, resilience is expected to be a system property that a system can have against all combinations of disturbance types and origins and its boundaries should cover the full spectrum, as shown in Figure 37.

Taking into account the fact that system resilience is a property and can be described by a set of systems characteristics, there has been an attempt to extend the definition of resilience in way that it covers all three types of survivability. For the necessary brainstorming process, system features that would be effective in all types of survivability were considered, with a notional example of human health response also used as an analog for importing ideas from resilient systems in biology. Table 7 contains a list of the proposed characteristics of a system that can be referred to as resilient, along with the corresponding functionality of two example systems, a naval system and the health response of the human body, maybe the most resilient system that can be found today in nature.

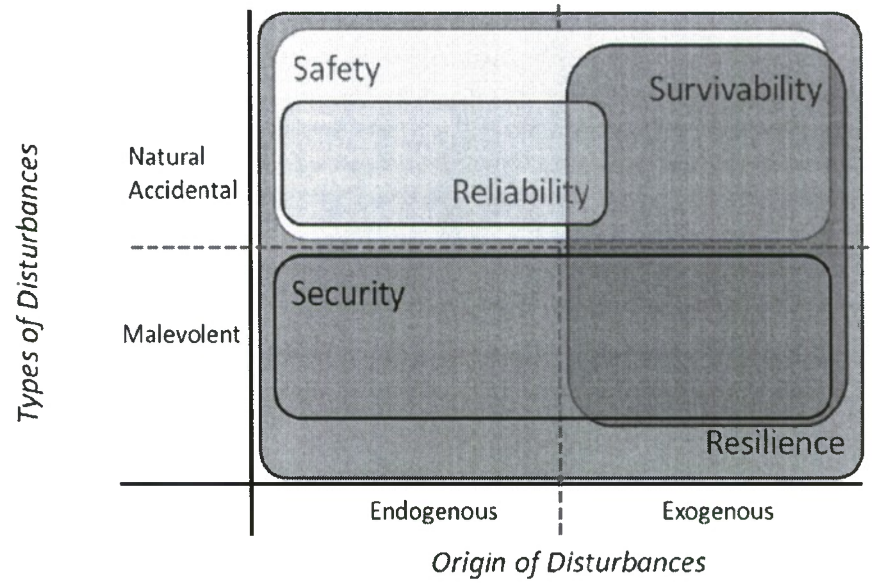


**Figure 37: Boundaries of Safety, Security, Survivability and Reliability, including Resilience**

**Table 7: Proposed extension of resilience definition, along with two illustrating examples**

| Survivability type | Resilience definition | Characteristic | Engineering System (e.g. threat missile attack) | Human Body (e.g. threat = flu) |
|---|---|---|---|---|
| Susceptibility | Extended to systems engineering | Ability to monitor threat | Use Radar detection to locate origin of missile and direction | Be aware of people with flu symptoms around you |
| | | Ability to sense threat | Estimate speed of missile and direction | Detect for flu symptoms on your body (sore throat, intense cough, etc) |
| | | Ability to warn about threat | Activate warning indications of display panel if missile headed towards ship | Change body temperature, body fluids, muscle aches, headache, fatigue |
| | | Ability to adapt to change for more effective system persistence to (adverse) change and system recoverability | Reposition ship to threat, maneuver away, activate anti-missile defense systems | Change of body temperature, sweating, |
| vulnerability | Basic Resilience definition | Ability to persist to change | Activate missile defense systems, utilize performance for avoiding or mitigating change | Activate mechanisms that fight against virus effects and spreading, enhancing immunity |
| | | Ability to absorb change | Shielding, material absorbing ability, compartmentation, fault isolation | Virus isolation, flu shot effects, finite recovery time |

| | | | | |
|---|---|---|---|---|
| Recoverability | Basic Resilience definition | Ability to maintain subsystem connectivity and integrity | Apply system redundancy, reconfigurability | Basic functions maintained through redundant organs (e.g., if cannot breath from nose, then use mouth) |
| | Extended to systems engineering | Ability to prevent system loss and fully recover the system's mission | Recollect and redefine mission objectives that are achievable through the new configuration (mission flexibility) | Be able to perform basic functions, vital body functions unaffected by virus spread, |

## ***Framework for System Resilience Assessment***

After a thorough investigation for State-of-the-Art assessment methods in the field of resilience engineering, it has been observed that the scientific domain is still at its infancy stage and researchers of this community are still proposing ideas for what would be an appropriate framework for resilience assessment, thus confirming that no analytical or quantitative framework for assessing system resilience exists yet. In fact, scientists have not even agreed on a standard and universally acceptable definition for what a resilient system actually is and what it does. Several definitions on what system resilience is are suggested, or even how a resilient system behaves, yet there has not been a clarifying set of universal definitions on system resilience as of present. Some entities have attempted to propose metrics in order to form a more formal framework, yet the latter ones have usually been formulated based on unknown or unclear criteria and without any demonstration of their effectiveness. Therefore, the main goal around this effort, is to develop a framework for quantitative system resilience assessment and demonstrate the effectiveness of the selected metrics.

To break the above objective further down, the following questions need to be addressed and lead us to answers:

- What is a unified definition of the following terms related to resilience: Resilience Engineering, resilient systems and their characteristics?
- What are the underlying assumptions for the previous definitions with respect to the system, its environment and its mission requirements?
- How is this definition different than earlier attempts to define system resilience that were more relevant to system safety, reliability, survivability, security, robustness?
- Are there metrics from existing assessment frameworks that could also be appropriate for resilience assessment, based on a given definition and assumptions?
- How could the Goal Question Metric (GQM) method by INCOSE be applied to generate metrics, based on characteristics of resilient systems and their typical expected behavior?
- With a given set of metrics, is there a formal analytical procedure that could be used to demonstrate the goodness of the selected metrics?
- If so, what figures of merit for metric goodness can be employed?
- Is a sensitivity analysis adequate, or more tests would be required to demonstrate the goodness of the metrics?
- Can resiliency characteristics be revealed by the extended framework?

If an existing analytical framework for survivability assessment is extended to account for dynamic system responses, including the effects from emerging behaviors and changing operating conditions and requirements, then it would adhere to the premises of resilience engineering and support the evaluation of system resilience. The central hypothesis of this task has been formulated and states the following:

*A more resilient system demonstrates improved survivability than a robust system, mainly by incorporating engineering system reconfigurability, if subject to the same intelligent or natural events that affect system operations. However, it should be expected that improved safety and survivability come at some expense in overall system performance, acquisition and maintenance costs.*

For supporting the above hypothesis, the objective is to develop and optimize two system architectures, one to exhibit the features of a robust system design and the other to be the resilient system. In order for this hypothesis to be tested and supported, not only a design approach is required for the acquisition of the two solutions, but also a complete evaluation framework that will include the necessary metrics for confirming that a solution is resilient, according to the previously discussed concepts and premises of resilience engineering.

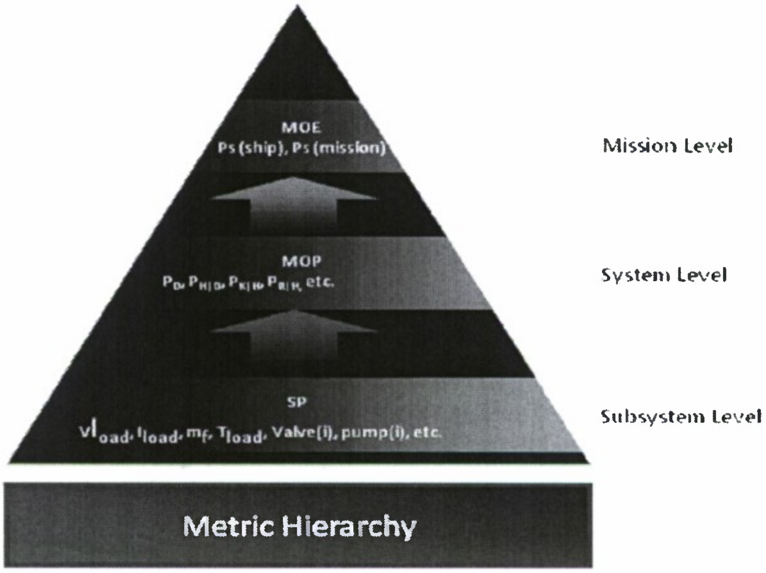
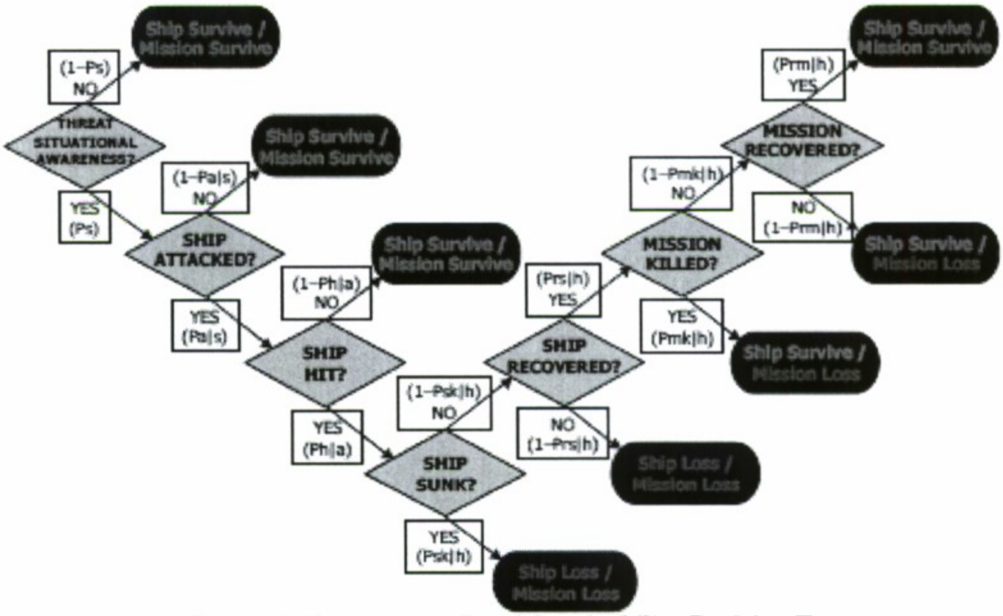

MOE
Ps (ship), Ps (mission)
Mission Level
MOP
$P_D, P_{H|D}, P_{K|H}, P_{R|H}$, etc.
System Level
SP
$V_{load}, I_{load}, m_f, T_{load}$, Valve(i), pump(i), etc.
Subsystem Level
Metric Hierarchy

**Figure 38: Metric hierarchy for survivability assessment framework**

Given that the ultimate objective of developing resilient systems is to improve overall survivability and mission effectiveness, a convenient starting point for the framework development would be one of the existing state-of practice methods for survivability assessment. One of the most common is the Total Ship Survivability Assessment Method (TSSA) [13]. There is a certain hierarchical distribution of metrics that allow for total ship and mission probability of survival calculations (

Figure **38**). At the lower level there are the subsystem metrics, or *System Parameters*. The next level includes the measures of performance, or MOPs that involve conditional probability calculations based upon the values of the SPs. The higher level is represented by the MOEs, including the aggregate metrics for high level mission and system survival assessment.

TSSA relies upon the concept of the *Kill Chain*. The entire incident is broken down into subsequent time epochs, at the end of each one; there is always an event with a set of possible outcomes. Calculation of the conditional probabilities for each event outcome is necessary for the MOE aggregate metric estimation for the total probabilities of mission *P(MissionLoss)* and system *P(SystemLoss)*survival. An example of a kill chain is depicted in Figure 39. As the transition from MOP to MOE seems more straightforward, the real challenge at this time is to develop, evaluate and select the appropriate SPs and effectively convert them into the corresponding MOPs.



Equation 1.

$$P(ShipLoss) = 1 - P_s P_{a|s}\left[1 - \sum_{i=0}^{i_{max}} P_{sh|a}\left(1 - P_{sk|sh}\right) P_{rs|sh}\right]$$

Equation 2.

$$P(MissionLoss) = 1 - P_s P_{a|s}\left[1 - \sum_{i=0}^{i_{max}} P_{sh|a} P_{rs|sh}\left(1 - P_{mk|sh}\right) P_{rm|sh}\right]$$

**Figure 39: Kill Chain and linking of the MOPs to the MOEs [13]**

The basic system information that this framework is called to be able to describe, should contain the following:

- System Geometry and Specifications

- Engineering Subsystems
    - Performance ratings
    - Cost
    - Connectivity
    - System states
- Mission profiles
    - Goals and objectives
    - Figures of merit
    - Time frames
- Threats and hazards
    - Types
    - Impact data and models
    - Failure rates and modes
    - Response and recovery times

The above requirements for metrics development are quite generic and there is a need at this point for a systematic procedure that will lead to a set of metric alternatives. This process will be based on associating the requirements to the metrics. One possible approach is to apply the "Goal-Question-Metric" process as suggested by INCOSE. Ideally, stakeholders and consumers would pose the goal and the question, and engineers would define the metric.

### *Demonstrating the GQM process*

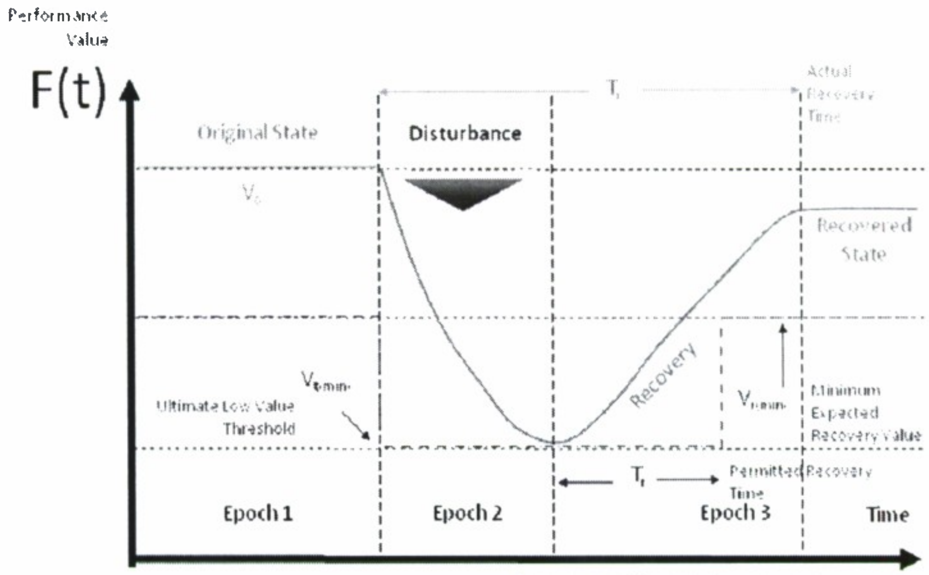
Here is a simple example on how the GQM process can be applied: Let's assume that the goal imposed by the stakeholder is to "increase product reliability". As a consequence, a relevant question can be formulated as follows: "What is the current fault removal rate compared to earlier releases of this product"? One metric that answers to the previous question is the current percent and number of faults removed by lifecycle phase and fault severity for this product release. Another possible metric is the previous percent and number of faults removed by lifecycle phase and fault severity for earlier releases.

There are four main categories under which every metric can be classified. According to its mathematical description, a metric can be at least one of the following:

- **Ratio**: We divide one quantity over the other, with the numerator and denominator are mutually exclusive
- **Proportion**: We divide one quantity over another, with the numerator and denominator are not mutually exclusive and numerator is part of the denominator
- **Percentage**: It is a conversion of a proportion in terms of –per hundred- units
- **Rate**: A rate represents the dynamic rate of change of the phenomena of interest over time

Returning to the dynamic behavior of a system with respect to its survivability, a generalized response can be captured by Figure 40. According to this definition as provided by the above figure, there can be at least two metrics of interest derived. Following the GQM approach, the two goals of the system can be:

- G1: Improve ability to minimize utility loss
- G2: Improve ability to meet critical utility thresholds



Based on definition formulated by [Richards 2003]

**Figure 40: Generalized value-based survivability definition [12]**

Two questions can be formulated as an attempt to explore possible metrics for the measuring of the system's ability to satisfy the two previous goals:

- G1→ Q1.1: What is the utility loss due to performance degradation
- G1→ Q1.2: What is a time dependent average measure of the utility loss due to performance degradation
- G2 → Q2: To what extent is the threshold satisfied, even after significant performance degradation?

At this point, metrics can be formulated as possible answers to the goals and questions:

- G1→ Q1.1→ M1.1: The utility loss due to performance degradation can be expressed as

$$U_L = U_0 - U(t) \tag{1}$$

- G1→ Q1.2: → M1.2: The time weighted average utility can provide a cumulative time dependent measure of the system's response due to performance degradation

$$\bar{U_T} = \frac{1}{T}\int U(t)dt \tag{2}$$

with $\bar{U_L} = U_0 - \bar{U_T}$ as the time weighted average utility loss

- G2 → Q2 → M2: Threshold availability $A_T$, where, TAT is the total Time Above Threshold

$$A_T = \frac{TAT}{T} \tag{3}$$

***<u>The proposed set of metrics for resilience assessment</u>***

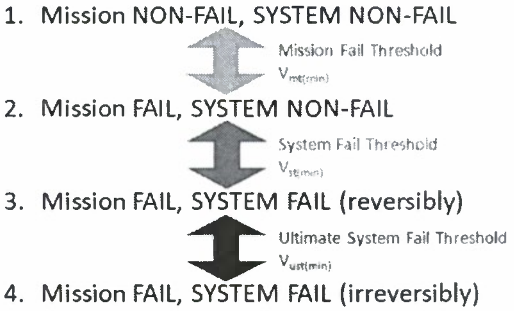
These metrics can also be attributed to describe how robust is the design, given that robustness is the Insensitivity of system value delivery to changing contexts [12] However, there are other characteristics of a resilient system that cannot be captured by metrics that mostly refer to system robustness. For instance, adaptability to changing mission requirements is not explicitly reflected, thus there is need for an extended metric set that will seek to address all the other features that can make a robust design to become more resilient.

One thought is to distinguish effects based on their origin. While the system is suffering from sudden performance degradation, immediately there can be two types of change identified: Change due to disturbance against value delivery (adversary) and change due mission updates or system reconfigurability (favorable). A change can be also described as a time dependent rate; therefore in this case there can be two time dependent rates of change that describe opposing actions. A total rate will capture total system ability to absorb change.

Except for balanced rates and time weighted value differences, the resilience characteristic of maintaining system shape and status can command for metrics that display a count of available entities or their health status. For instance, the proportion:

$$S = \frac{N_{Existing\ Connections}}{N_{Commanded\ Connections}} \tag{4}$$

is providing a ratio that works as a shape factor, which describes both component multitude and availability.



1. Mission NON-FAIL, SYSTEM NON-FAIL
2. Mission FAIL, SYSTEM NON-FAIL
3. Mission FAIL, SYSTEM FAIL (reversibly)
4. Mission FAIL, SYSTEM FAIL (irreversibly)

**Figure 41: Thresholds of transitioning through Mission-System States**

With the proposed set of characteristics of resilient systems as defined by Table 7, the application of the GQM method becomes more straightforward. Each characteristic can be assigned to a set of goals that would become what the designer should have in mind if he was about to design a resilient system. A design method requires a set of metrics and measures to be available, in order to evaluate the merits of a solution and allowing the designer to perform the necessary tradeoffs towards his effort to identify the solution that is feasible at a minimum cost.

In order to identify the goals, one needs to think beyond the properties of a resilient system and conceptualize the dynamic behavior of the system in some common disturbances that it can experience. Given that the effectiveness of a system is mainly determined by its ability to perform its mission and its overall health and integrity, combinations of states regarding its availability and mission effectiveness can be devised. Figure 41 displays 4 combinations of states, with the last two only differing in the system's ability to reverse its status or not being able to recover at all. For each one of the four status cases, notional time histories have been constructed that demonstrate the alternative paths that a system can follow regarding its dynamic performance and output value delivery.

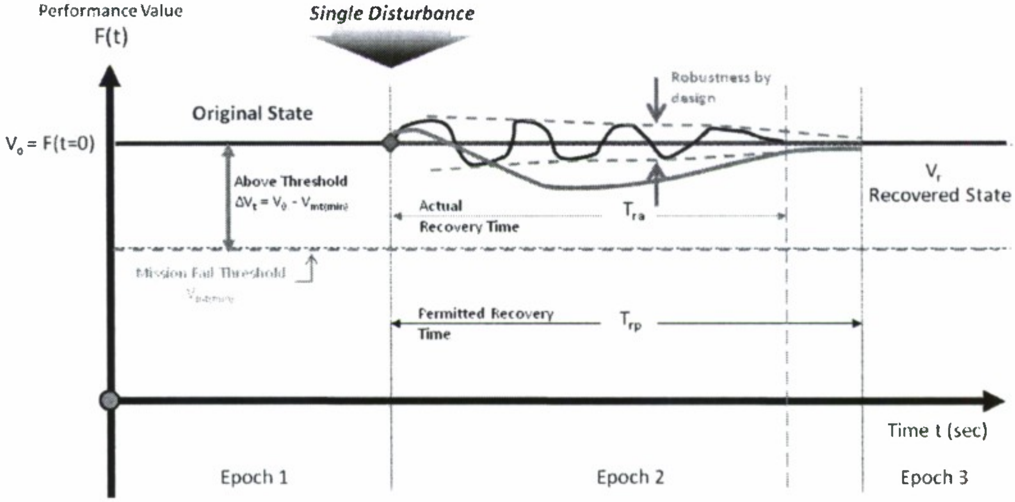


**Figure 42: Case 1 – Mission NON-FAIL with System NON-FAIL**

For the first case, the system maintains its mission performing ability along with its integrity. This is the ideal case where the effects of the disturbance are either weakening until they reach the system, or the system's inherent robustness manages to neutralize the effects and become insensitive to the threat. While this is a feature of a resilient system, it is not the sole defining characteristic. Figure 42 shows a notional time history of how such a system would behave over time.

For case 2, the corresponding behavior is displayed by Figure 43. The system's performance output is degraded and reaches a minimum point below the mission threshold and there may be an increased risk of not maintaining its physical integrity. Enhancing the system's survivability by incorporating technologies that can at least prevent the system from losing its integrity is a commonly used approach when doing robust design of systems. However, it is seldom that such enhancements will be considered in the early design phases and they are usually incorporated by retrofitting on the original design and by making the appropriate modifications for their seamless integration on the design. Most of the time this approach is responsible for extra weight and cost, and not always the effectiveness of the solution is guaranteed, especially under unknown or unconsidered mission conditions.

A resilient system would maintain the same goal of not allowing itself to degrade to the point that its integrity might be lost, yet it also carries by design, a set of alternative internal mechanisms that perform some extra functions compared to the robust system in order to achieve the same result. In other words, it should actively sense the threat approaching and begin to adapt to the change that is about to occur. That requires intelligent decision making in order to reach a state where during the degradation phase it can reconfigure effectively and have all the resources available for supporting its recovery at a minimum time period.

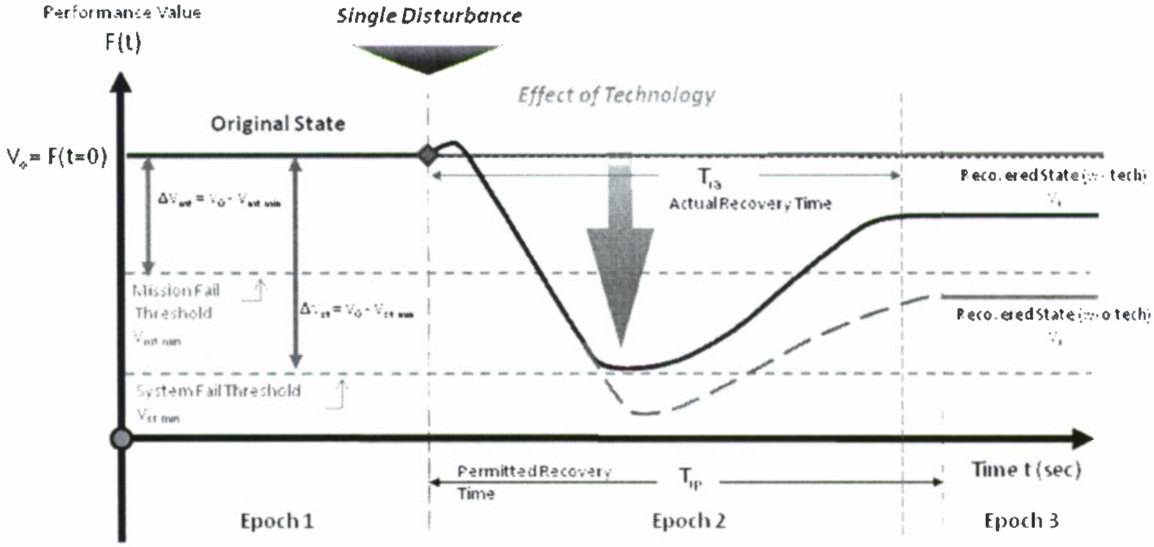


Figure 43: Case 2 – Mission FAIL with System NON-FAIL

If the system either naturally, or because it's a resilient design can follow the procedures described above, then it should be capable of not reaching the ultimate system fail threshold, thus its degradation to be temporary and reversible. As it is shown in Figure 44, not only it recovers above the system fail threshold, thus regaining its integrity, it should also restore its mission ability and hopefully to restore its value delivery back to the original state that it maintained before it experienced the disturbance.

On the contrary, a system that is not resilient, would fail to maintain itself above the ultimate system fail threshold and irreversibly bring itself into a state where it cannot either perform its basic functions or maintain its integrity and become bound to a catastrophic failure, as shown in Figure 45.

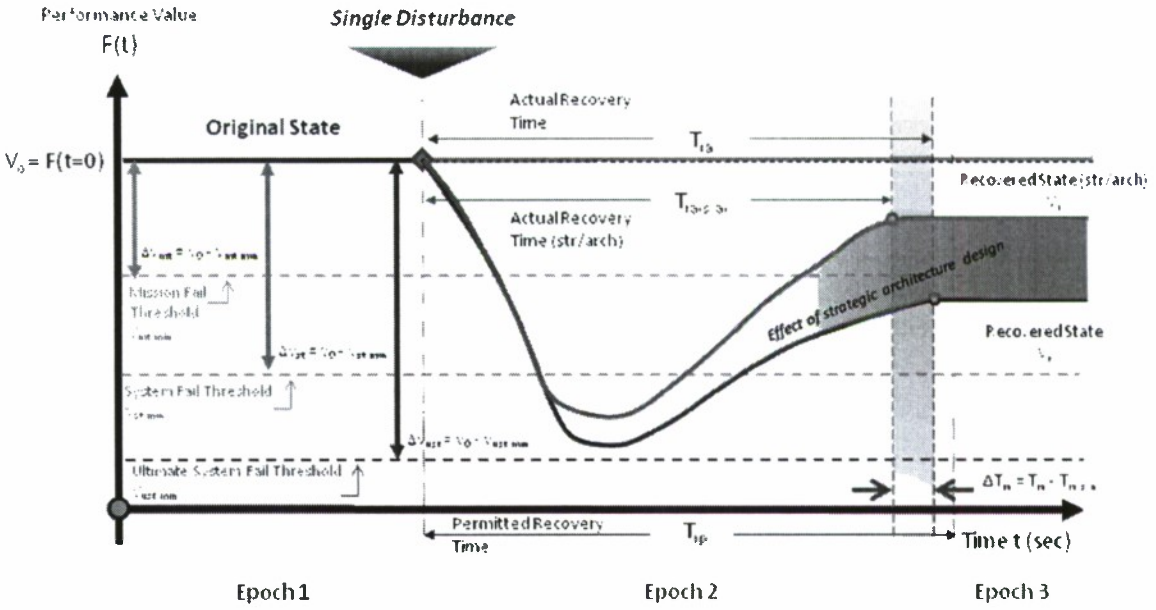


**Figure 44: Case 3 – Mission FAIL with System FAIL (Reversibly)**

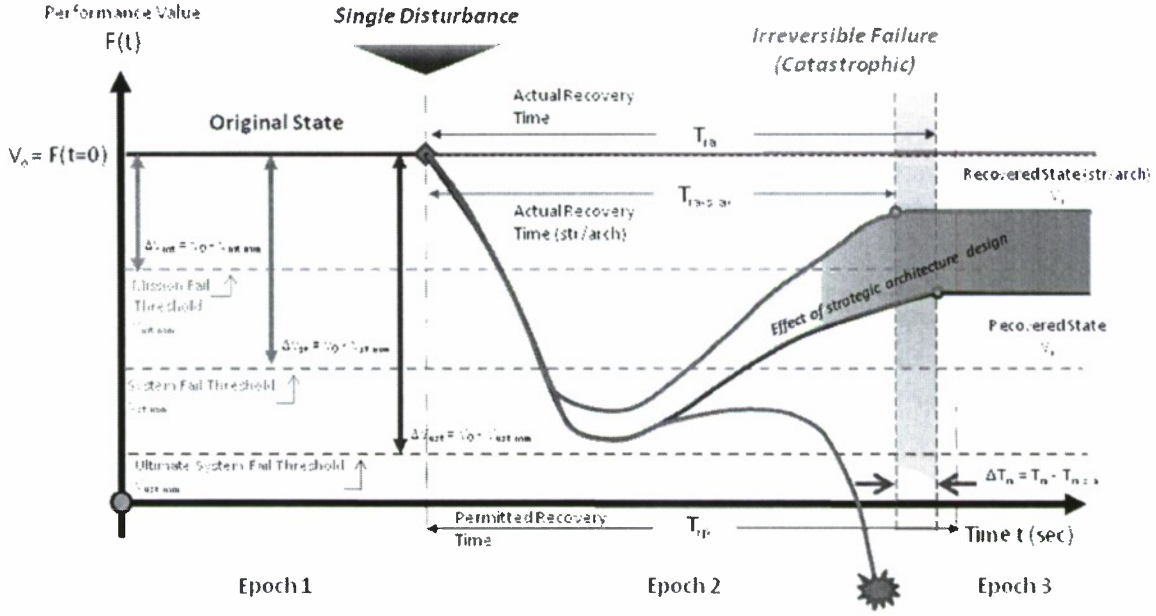


**Figure 45: Case 4 – Mission FAIL with System FAIL (Irreversibly)**

By taking into consideration how a resilient system would behave according to the previous illustration and combining that to GQM approach, a thorough set of metrics has been proposed and is presented in Table 8. It is hypothesized that the metrics can capture all aspects of how a resilient system should behave in order to support its mission and its recovery if degraded. In order to support this hypothesis and demonstrate the goodness of the proposed metrics, a simple example problem is currently under construction. Given that the concept of resilience also applies in materials, with elasticity and plasticity thresholds to play the same role as the system performance threshold, an analogy has been drawn and a cantilever beam with an actuating controller has been selected as a demonstration model. The plan is to test the sensitivity of the metrics for different configurations (baseline vs. robust vs. resilient), under the same scenario containing the same stimulant function and investigate how well the metrics can capture the improved performance of the more resilient configuration compared to the baseline.

***Experimental Setup***

The demonstration model involves a cantilevered beam that can experience continuous forces through a weight distribution F = w(x), along with impulsive instantaneous force signals ΔF at its edge, that typically last only for a small time period Δt. As a result of this input stimulus, the beam is experiencing a deflection from its original horizontal equilibrium position, along with a certain amount of curvature that changes its shape. The strain that the beam is experiencing and determines the deflection and the curvature are immediate functions of the beam's shape, material and the effect of any actuator control system that may be included in the experimental setup. The experimental setup is described by Figure 46.

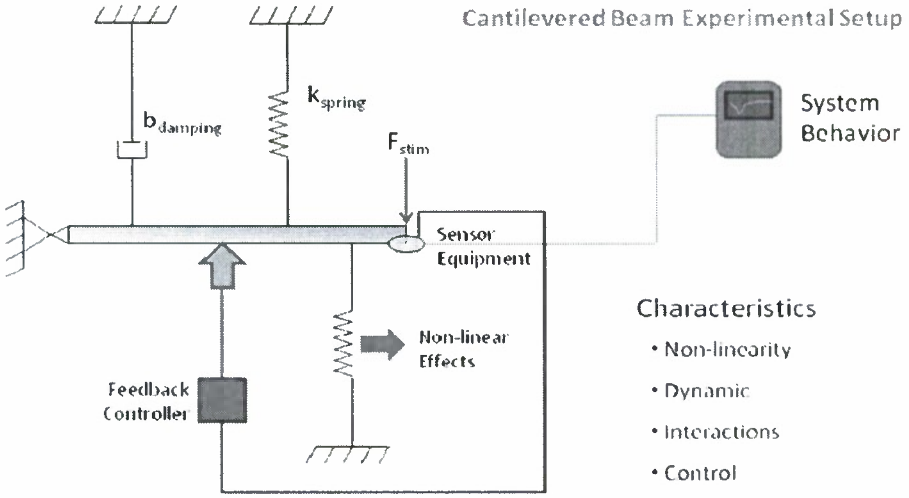


**Figure 46: Simple beam model for testing the proposed resilience assessment framework**

| Resilience Definition | Characteristic | Goal | Question | Metric |  |  |  |
|---|---|---|---|---|---|---|---|
|  |  |  |  | Description | Formula | Range | Unit |
| Extended to systems engineering | Ability to monitor threat | Increase the system's situational awareness | What is the percentage of threats, hazards and risks that can be monitored in a given range around the system? | **Detectability D**: Ratio of # of detected threats over the total # in a given range | $\boxed{D=\frac{N_{Detected}}{N_{Total}}\cdot 100(\%)}$ | 0-100% | ND |
|  |  |  | How does the effect of the threat affect the system's mission and health? | **Utility function U**: Specify system performance outputs and mission effectiveness metrics (U) | U = Power delivery to loads (P), Heat dissipation rates (Q_dot), stability metrics, etc. | Various | Various: Watts, Joules/sec, etc. |
|  | Ability to sense threat | Collect more information on the threat and predict its next actions | What is the rate at which the threat affects system performance? | **Degradation Rate DR**: Time derivative of performance value loss ΔU | $\boxed{DR=-\frac{dU}{dt}, U=U_0-U(t)}$ | DR:[D, $\Delta(U_{ust})/\Delta t$] | (U unit)/sec |
|  |  |  | How well can the system predict the effects of the threat and its upcoming performance degradation? | **Predicted Degradation Rate PDR**: Predicted DR | Actual DR by predicted DR, R^2 | 0-1 | ND |
|  | Ability to warn about threat | Increase the system's responsiveness after motoring a certain number of threats | What is the time required from monitoring the threat to sensing its effects? | **Responsiveness RT**: Min required time from monitor to sense over time period from monitor to sense | $\boxed{RT=\frac{\Delta t_{\min}}{(t_{Sense}-t_{Monitor})}}$ | RT: [1, inf] | ND |
|  | Ability to adapt to change for more effective system persistence to (adverse) change and system recoverability | Enhance system adaptability to changing conditions by shifting the thresholds that separate different system-mission states and thus enhance the system's tolerance to the effects of the threat | How do the mission and system performance thresholds change to reflect the system's accomodation? | **Threshold**:<br>Type1: Mission FAIL, System OK,<br>Type2: Mission FAIL, System PART FAIL,<br>Type 3: Mission FAIL, System TOTAL FAIL | $\boxed{\begin{array}{l} Type1:\Delta U_{mt}=U_0-U_{mt} \\ Type2:\Delta U_{st}=U_0-U_{st} \\ Type3:\Delta U_{ust}=U_0-U_{ust} \end{array}}$ | Various | Various: Watts, Joules/sec, etc. |
|  |  |  | What is the system's ability to meet critical thresholds? | **Threshold Availability $A_T$**: Total time above threshold over total time | $\boxed{A_T=\frac{TAT}{T}}$ | 0-1 | ND |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Basic Resilience definition | Ability to persist to change | Increase the system's capacity of fighting against the change imposed by the threat and minimize the effects of the changes occuring | What is the rate at which the system's performance is degrading? | **Degradation Rate DR**: Time derivative of performance value loss ΔU | $DR = -\frac{dU}{dt}, U = U_0 - U(t)$ | DR:[0, Δ(U_ust)/Δt] | (U unit)/sec |
| | | | What is the duration of the threat effects? | **Degradation Time period $\Delta t_{deg}$**: Time elapsed from start of disturbance effects $t_{deg0}$ until the max performance degradation from normal performance conditions $t_{Umin}$ | $\Delta t_{deg} = t_{U\min} - t_{deg0}$ | $\Delta t_{deg}$: [0, inf] | sec |
| | | | What is the max performance degradation? | **Max performance degradation $\Delta U_{max}$**: Max difference between original performance value $U_0$ and U(t) | $\Delta U_{max} = U_0 - U(t)_{mi}$ | $\Delta U_{max}$: [0, inf] | (U unit)/sec |
| | | | What is the improvement (reduction) on the DR, for a more resilient architecture? | **ρ-factor**: Coefficient on the time derivative of performance value loss ΔU, representing reconfigurability effects | $DR_p = -\rho\frac{dU}{dt}, U = U_0 - U(t)$ | ρ:[0, 1] | ND |
| | Ability to absorb change | If threat cannot be eliminated, Increase the system's ability to become insensitive to the unavoidable change imposed by the effects of the threat | What is the improvement (decrease) on the performance degradation , for a more resilient architecture? | **b**: Decrease on the performance value loss ΔU, representing reconfigurability effects | $\Delta U'_{max} = \Delta U_{max} - b = [U_0 - U(t)_{min}] - b$<br>$\Delta U'_{max} = U_0 - [U(t)_{min} + b]$ | | (U unit)/sec |
| | | | What is the average loss of system performance due to the effects of the threat? | **Time weighted average performance loss $U_L$: Difference between the normal** performance output minus the time weighted average performance of the system during the degradation | $\bar{U}_T = \frac{1}{T}\int U(t)\,dt$<br>$\bar{U}_L = U_0 - \bar{U}_T$ | | (U unit)/sec |
| | | | What is the relative loss compared to the threat effects? | Signal-to-Noise Ratio S/N: Performance output U(t) over the time weighted average performance loss $U_L$ | $S/N = \frac{U(t)}{\bar{U}_L}, S/N_{log} = -10 * \log_{10}(S/N)$ | | ND |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Basic Resilience definition | Ability to maintain subsystem connectivity and integrity | Maintain all subsystem good health status and necessary connections amongst them | What is the loss ratio for subsystems? | **Subsystem Loss Ratio**: Number of damaged/inoperable subsystems over total initial number of subsystems | $\boxed{SLR = \frac{N_{Damaged}}{N_{Total}} \cdot 100\,(\%)}$ | 0-100% | ND |
| | | | What is the loss in subsystem connections? | **Connection Loss Ratio**: Number of damaged/inoperable connections over total initial number of connections | $\boxed{CLR = \frac{N_{Damaged}}{N_{Total}} \cdot 100\,(\%)}$ | 0-100% | ND |
| Extended to systems engineering | Ability to prevent system loss and fully recover the system's mission | Maintain system health and mission effectiveness metrics close to their target values | What is the recovery rate while system performance is being restored? | **Degradation Rate DR**: Time derivative of performance value loss ΔU | $\boxed{RR = \frac{dU\ (t)}{dt}}$ | DR:[0, inf] | (U unit)/sec |
| | | | What is the min time required to reach a steady state? | **Minimum Recovery time $t_{rp}$**: Time elapsed from performance min up to SS recovery | $\boxed{\Delta t_{rp} = t_{ss} - t_{U\min}}$ | $\Delta t_{rp}$: [0, inf] | sec |
| | | | How off is the restored state from the original? | **Recovery Offset ΔUrec**: Difference between original performance value $U_0$ and $U_{rec}$ | $\boxed{\Delta U_{rec} = U_{ss} - U_0}$ | $\Delta U_{rec}$: [0, inf] | (U unit)/sec |
| | | | What threshold are/are not satisfied and what is the offset? | **Threshold Offsets**:<br>Type1: Mission FAIL, System OK,<br>Type2: Mission FAIL, System PART FAIL,<br>Type 3: Mission FAIL, System TOTAL FAIL | $\boxed{\begin{array}{l} Type1: \Delta U_{mt} = U_{ss} - U_{mt} \\ Type2: \Delta U_{st} = U_{ss} - U_{st} \\ Type3: \Delta U_{ust} = U_{ss} - U_{ust} \end{array}}$ | arious | Various: Watts, Joules/sec, etc. |
| | | | What is the total recovery time? | **Total Recovery time $t_{rt}$**: Total degradation time plus total recovery time | $\boxed{\Delta t_{rt} = \Delta t_{\deg} + \Delta t_{rp}}$ | $\Delta t_{rt}$: [0, Inf] | sec |

Table 8: Proposed resilience assessment framework by applying the GQM method

***Formulation of analysis tools and integration into a M&S environment***

The second main objective of this task is to demonstrate the need for a damage modeling and identification tool, which has the ability to identify emerging behaviors and "hidden" modes of failure in addition to the SoA common cause failure identification approaches. A dynamic simulation of the system's operations is required for allowing for identification of the extra risks. The idea is to combine a fault tree analysis tool such as DOMINO to the dynamic M&S and have them exchange data at the end of a simulation time step. The main hypothesis here is that the combination of both tools should be able to reveal more sources of possible failure or disruption, thus bring more insight regarding the system's operation and provide extra evidence as to what additional enhancements would be required to make it a more resilient design.

Regarding the application of the resilience assessment framework in a naval system related problem, the original plan was to combine individual models of engineering subsystems to produce integrated models for dynamic simulation. The most significant part of this particular effort will be a routine that models and investigates damage propagation on a naval system. The damage model engine will analyze (damage prediction) and visualize (on the Paramarine ship model) the damage propagation throughout the particular architecture. In this task, a total ship systems operations M&S environment is the desired outcome, including an investigation of damage generation and propagation. An overview of this attempt is given by Figure 47.

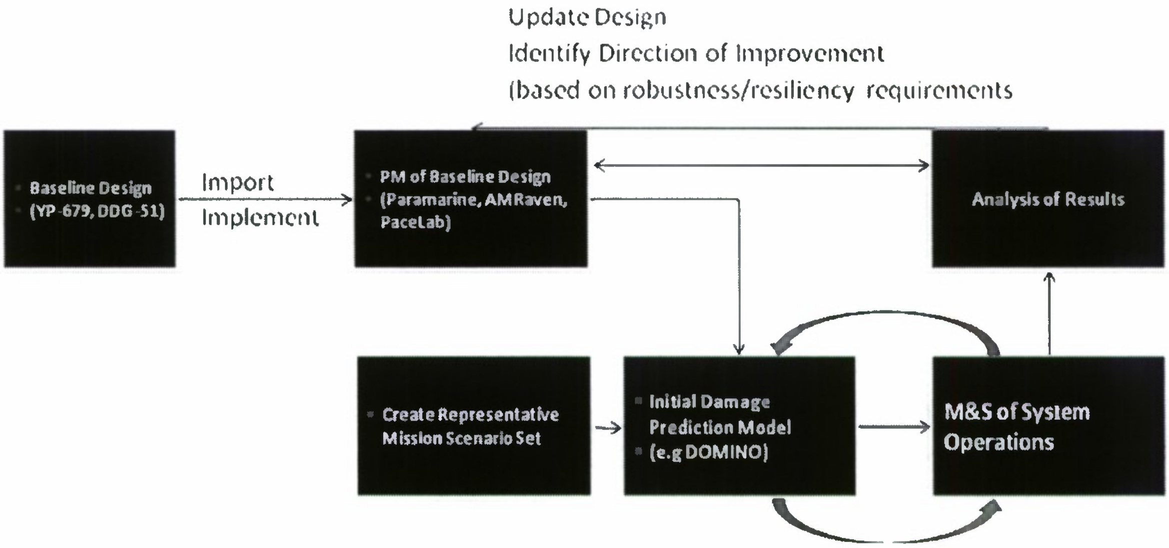

Update Design
Identify Direction of Improvement
(based on robustness/resiliency requirements
Baseline Design
(YP-679, DDG-51)
Import
Implement
PM of Baseline Design
(Paramarine, AMRaven, PaceLab)
Analysis of Results
Create Representative Mission Scenario Set
Initial Damage Prediction Model
(e.g DOMINO)
M&S of System Operations

**Figure 47: Modeling &Simulation Environment for Survivability Analysis**

As a baseline, a notional naval ship design is required to be the starting point for the implementation of the method. At this point, the baseline design of the YP ship has been finalized. Despite the availability of the YP ship model, including its two alternative configurations that are currently under work, there have been some thoughts for a larger baseline ship architecture design. Regarding the task (Task 5) of developing a survivability-based design method, it appears that a small scale ship, such as the baseline YP might not be sufficient for conducting adequate survivability studies. A larger architecture is expected to offer more meaningful results when running a typical damage scenario, with damage propagation extended throughout the ship to a certain extent, allowing for cases where the ship can still remain partially intact. There are definitely doubts that the YP architecture design might just suffer a total catastrophic failure from a single missile attack, given the fewer subsystem zones and limited available reconfigurability strategies for improving survival.

It requires a certain amount of information for the creation of a ship baseline, such as ship geometry, engineering subsystems, acquisition and operations cost breakdown, mission profiles, threats and hazards and local environmental conditions. Most of this information is not available, therefore a large amount of assumptions is being made and configurations are being formulated according to prior knowledge about similar ships or engineering intuition. This will be the case for the engineering plant that is a modified version of an IPS configuration.

The initial damage prediction module has been provided by the Navy (DOMINO) and it is now understood how this enabler can be integrated into the modeling process. Within DOMINO a connectivity schematic of the systems architecture is being created, in order to specify damage propagation due to initial connectivity. At the end of every time step of the systems simulation, DOMINO will be exchanging information with the dynamic simulation module and feeding this back to Paramarine, in order to conduct other static analyses of interest.

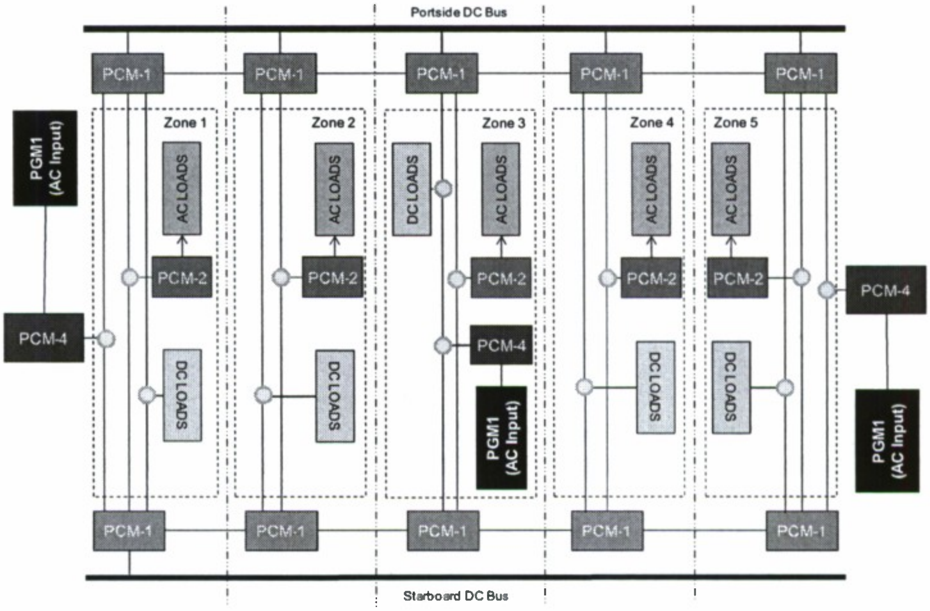


**Figure 48: Modified IPS architecture for DC electrical system M&S (based on Fireman & Doerry)**

The engineering system architecture is heavily based upon the IPS architecture, as presented by Fireman & Doerry [6] with four zones refers to a DC electrical power distribution system and is described in Figure 48. The most significant changes on this IPS original architecture were the addition of an extra zone to conform to the size of the enlarged baseline of the ship and the addition of DC or AC load placeholders to all zones in order to be able to size accordingly at the time that the design method will return the most resilient solution.

In terms of sourcing components to build the computational M&S, the main donor will be the Tabletop 2.0 simulator, with the incorporation of a set of power system components that ASDL had developed in the past. Reconfigurability strategies and enablers are borrowed from the ZELDA model that was made available to ASDL by Anteon Corporation through ONR. Finalizing the M&S baseline and integrating it with DOMINO is currently under construction and is expected to become a key enabler for improving the system's reconfigurability and adaptability to failures and disruptions.

### Future Work

Under the assumption that the framework for resilience assessment is finalized along with the M&S capability for damage and failure identification, the last part of this effort is to bring these enablers into a complete design methodology that will support the design and development of more resilient systems. Previously, a planned exercise has been considered for making a case regarding the benefits of resilient systems against designs that are suggested from SoA approaches. It is expected that while resilient systems should be more mission capable and survivable, this would not come at no cost and thus a cost effectiveness study would offer more insight as to whether the concepts of resilience engineering should become the paradigm for more safe and survivable designs.

## Publications

1. Michael Balchanos, Yongchang Li, Dimitri Mavris, "A Theoretical Framework for the Analysis and Design of Resilient Engineering Systems", To be submitted to the Journal for Safety Research, by Elsevier, 2010 (In progress)
2. Zhang, D., Balestrini, S., Weston, N., Mavris, D., "Distributed Online and Automatic Probabilistic Inference for Large Scale Complex Systems", Probabilistic Engineering Mechanics, Submitted in Dec, 2009. (Under review)
3. Kyungjin Moon, "Modeling and Simulation for Damage Analysis of Intelligent, Self-Reconfigurable Ship Fluid Systems in Early Design Phase, will be submitted to: Simulation Modelling Practice and Theory, Elsevier (In progress)

## References

[1] U.S. Navy, Yard Patrol Craft YP data sheet, available online at: http://www.dt.navy.mil/tot-shi-sys/com-cra/pro-gal/yp] , accessed October 2009.

[2] U.S Navy, Data Sheets on YP-679, available online at: http://www.navy.mil/navydata/], accessed October 2009.

[3] U.S. Naval Academy, "EE 301, Shipboard Power Distribution Systems", lecture notes, retrieved October 2009.

[4] Ball, R. E., "The Fundamentals of Aircraft Combat Survivability Analysis and Design", AIAA (American Institute of Aeronautics & Astronautics), 2003.

[5] Ball, R. E. & Atkinson, D. B.,"A History of the Survivability Design of Military Aircraft", *Naval Postgraduate School Report, AIAA,* 1998, 16.

[6] Doerry, N.H., H. Fireman, "Designing All Electric Ships," Presented at IMDC 2006, Ann Arbor, Michigan, 16-19 May 2006.

[7] Harriss, R., Hohenemser, C., Kates, R. , Goodman, G. & Rowe, W. *(ed.),* "Energy risk management", *Academic Press,* 1979, 103-138.

[8] Holling, C.,"Resilience and stability of ecological systems", Annual review of ecology and systematics, Annual Reviews, 1973, 4, 1-23.

[9] Hollnagel, E., Woods, D. & Leveson, N., "Resilience engineering: Concepts and precepts", Ashgate Pub Co, 2006.

[10] Dijkstra, A., Resilience Engineering and Safety Management Systems in Aviation, 2007.

[11] McManus H., Richards M., et. al, "A Framework for Incorporating "ilities" in Tradespace Studies", *AIAA Journal,* 2005.

[12] Richards, M., "Multi-Attribute Tradespace Exploration for Survivability", Engineering Systems Division, Massachusetts Institute of Technology, 2009.

[13] Yarbrough, N.R., Russell E.K., "The Joint Command and Control Ship (JCC(X)) Approach to Survivability Requirements Development: Total Ship Survivability Assessment," Association of Scientists and Engineers – 38th Annual Technical Symposium, May 9, 2002.

[14] Laracy, J., Leveson, N., "Applying STAMP to Critical Infrastructure Protection", 2007.

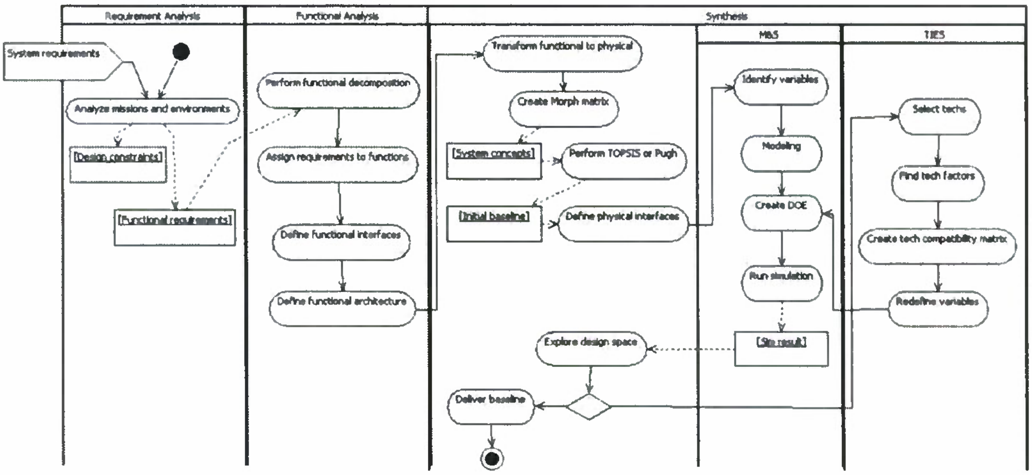
[15] Leveson N. L., "Role of Software in Spacecraft Accidents", *Journal Of Spacecraft And Rockets,* 2004, *41*, 564-575.

[16] Leveson, N., "White Paper on Approaches to Safety Engineering," 2003.

[17] Leveson, N., "Safeware: system safety and computers", *ACM New York, NY, USA,* 1995.

[18] Xiang, Y., Poole, D., Beddoes, M. P., 1993. Multiply sectioned Bayesian networks and junction forests for large knowledge-based systems. Computational Intelligence 19.

# Appendix



Based on "System Engineering Primer and Handbook.

**Figure 49: Activity Diagram for "Generate Concept"**

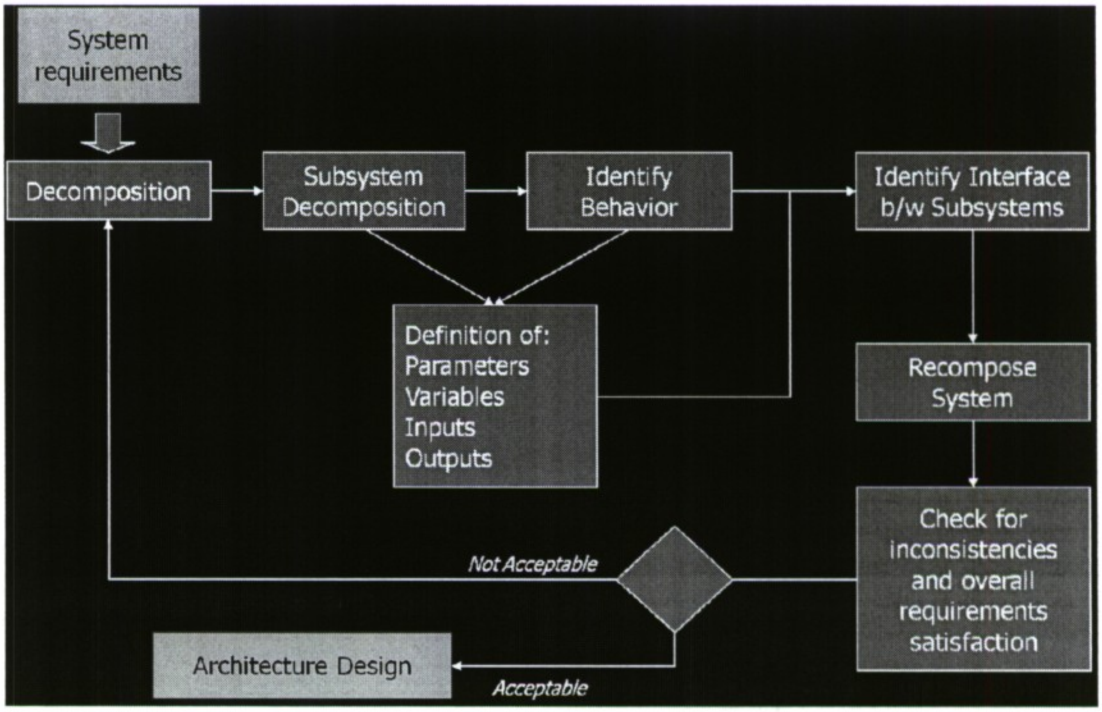


**Figure 50: Activity Diagram for "Define System Architecture"**

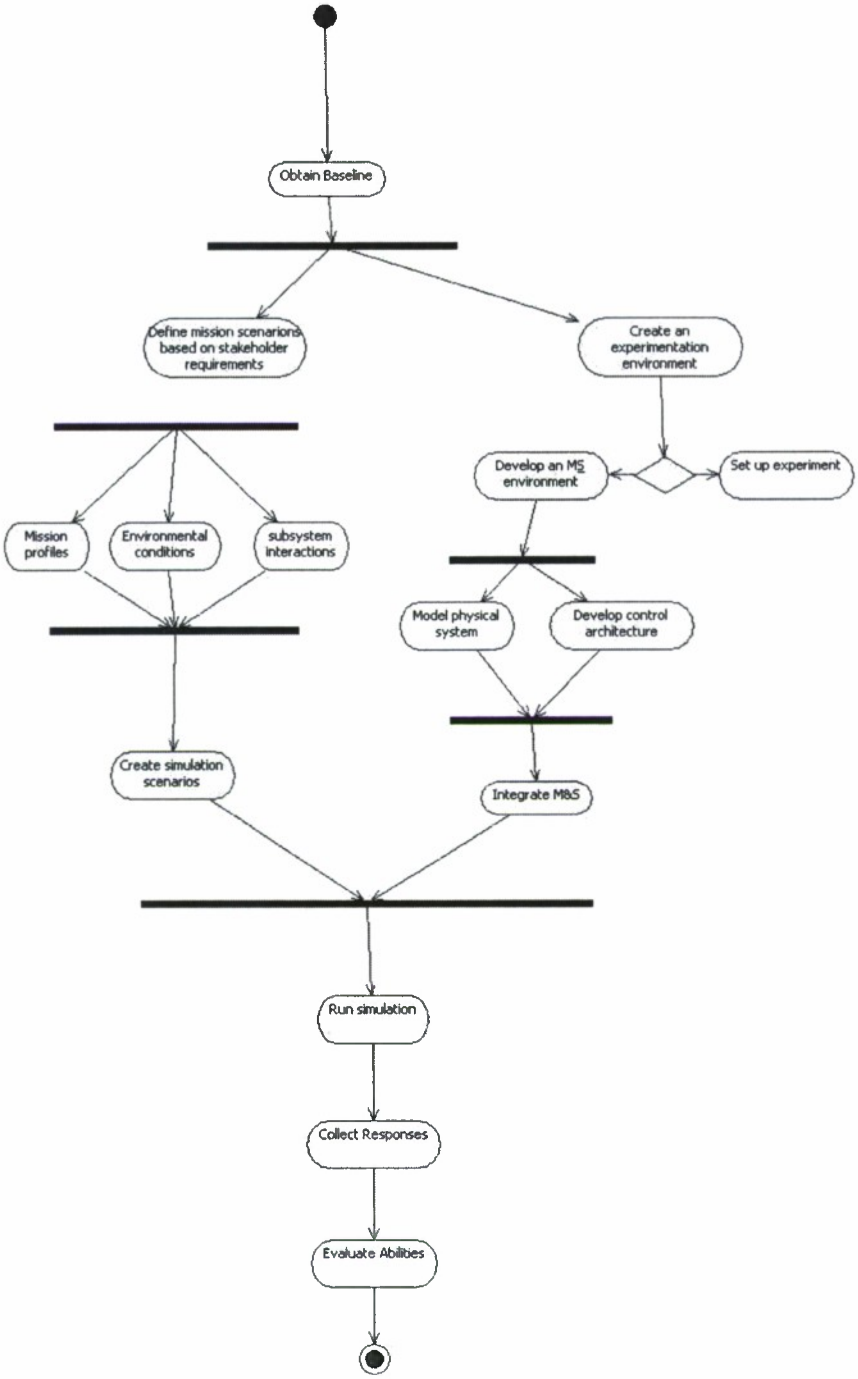


Figure 51: Activity Diagram for "Evaluate Ability"

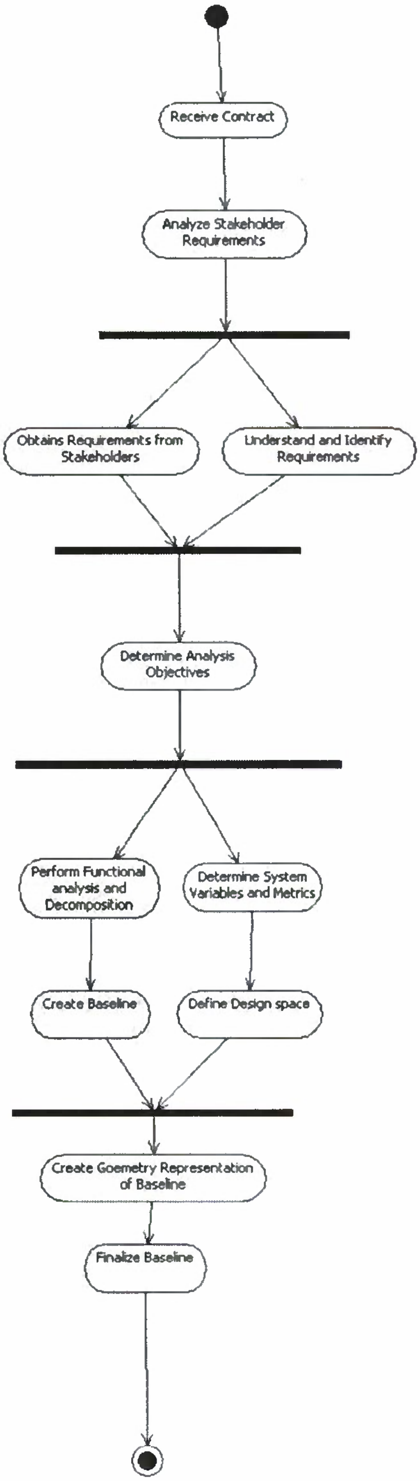

Receive Contract
Analyze Stakeholder Requirements
Obtains Requirements from Stakeholders
Understand and Identify Requirements
Determine Analysis Objectives
Perform Functional analysis and Decomposition
Determine System Variables and Metrics
Create Baseline
Define Design space
Create Goemetry Representation of Baseline
Finalize Baseline

**Figure 52: Activity Diagram for "Analyze System"**

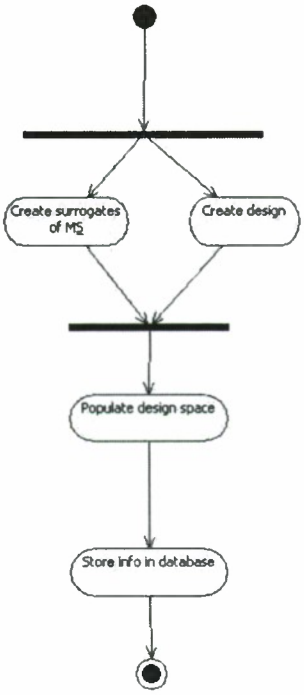
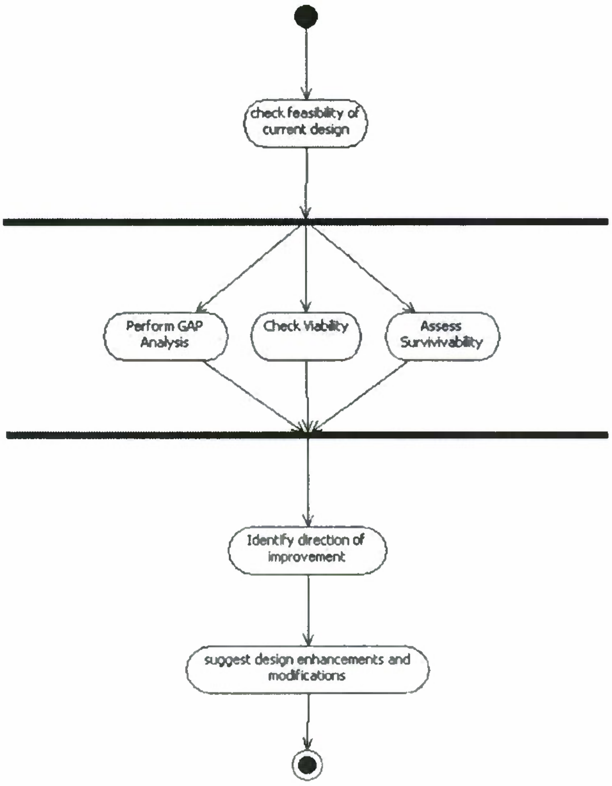
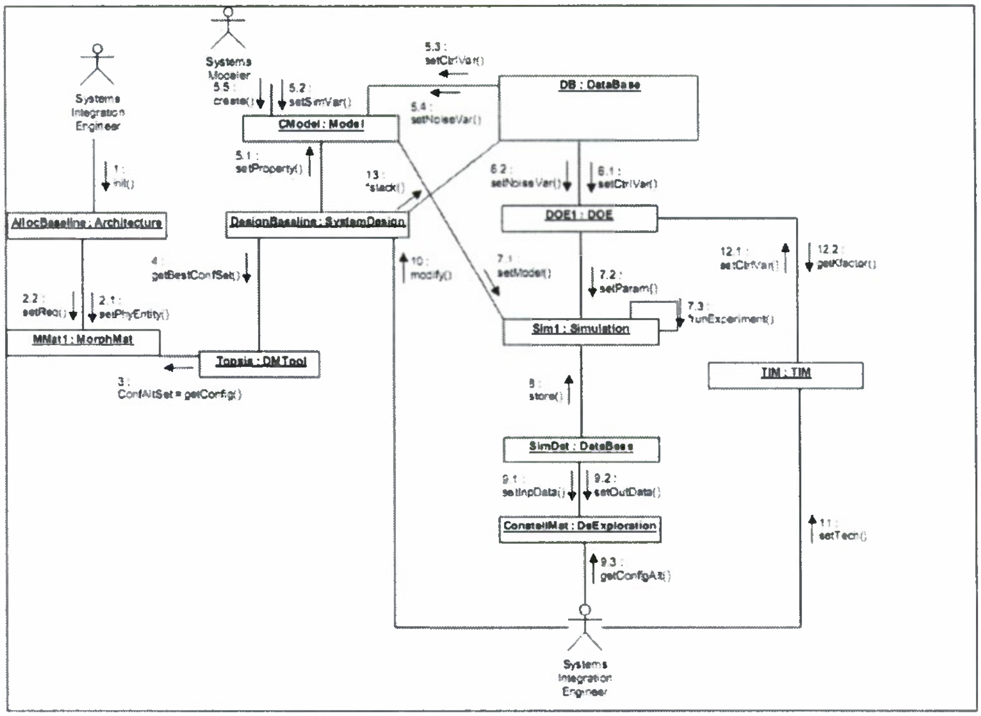


Figure 53: Activity Diagram for "Store Design"

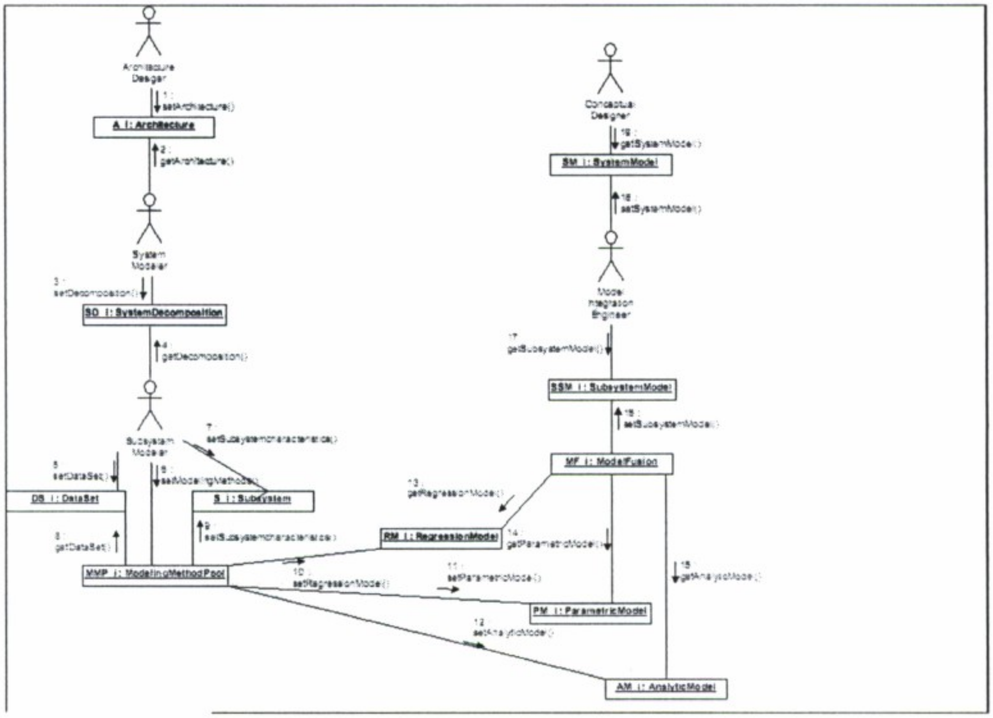


check feasibility of current design
Perform GAP Analysis
Check Viability
Assess Survivivability
Identify direction of improvement
suggest design enhancements and modifications

Figure 54: Activity Diagram for "Modify Design"



## Figure 55: Communication Diagram for "Generate Baseline"



## Figure 56: Communication Diagram for "Build System Model"